QLAVE

VOLUMEN V № 2 MARZO/ABRIL 1988

INFORMACION SOBRE Qlave

La integración en la asociación Qlave se hace por suscripción ANUAL.

El CEIUQL consta en el Registro Nacional de Asociaciones con el número 63210, y en el Registro Provincial de Zaragoza con el número 1742.

Qlave publica regularmente el boletín de los socios. Más información sobre la asociación puede obtenerse desde la SECRETARIA del Club.

Para ser miembro de Qlave se requiere estar interesado por el ordenador personal SINCLAIR QL.

El Club mantiene una LIBRERIA DE SOFTWARE.

La correspondencia debe enviarse al apartado de correos número 403/50080 de Zaragoza. Especificando si va dirigida a SECRETARIA, TESORERIA, LIBRERIA o si es una colaboración para publicarse en el Boletín.

GRUPOS LOCALES

ALBACETE	1	Rafael R. Bermudez (tlfno. 22 30 42).
ALICANTE	1	Blas Ortuño (tlfno. Zarandieta 7,4B).
GALICIA	1	Dasio Carballeira (tlfno. 381 56 74). 56 59 74
MADRID	1	Enrique Hernandez (tlfno. 4 03 01 99).
MALAGA	1	Salvador Merino (tlfno. 47 50 43).
MALLORCA	1	Pedro Egea (tlfno. 41 62 38).
NAVARRA	1	Jaime Lacasa (tlfno. 23 69 50).
SEVILLA	1	Rafael Candau (tlfno. 42 22 74) Jose M. Guzman (tlfno. 65 46 73).
VALENCIA	1	Enrique Sanchis (tlfno. 3 64 20 18).
VALLADOLID	1	Ignacio Enrique Cabero (tlfno. 35 59 82).
ZARAGOZA	1	Juan Alvarez (tlfno. 51 71 31) José Luis Fornies (tlfno. 35 54 69).

CONTRIBUCIONES AL BOLETIN QLAVE

Las contribuciones a Qlave deben ser ficheros QUILL en cartuchos de microdrives o en floppies de 3.5 preferiblemente. Las colaboraciones se devolveran a vuelta de correo. El número de lineas maximo por página es de 45 y el margen será a 80 columnas. Qlave no se hace responsable del contenido de los articulos firmados por su autor.

SE PROHIBE LA REPRODUCCION TOTAL O PARCIAL DEL CONTENIDO DEL BOLETIN

EDITORIAL

Convocada la Asamblea General Extraordinaria en el dia 13 de abril de 1988 en la Calle Mayor, y en el Mesón "El Carato" de Zaragoza, se reune la Junta directiva formada por los siguientes Srs.:

Presidente en funciones: Diego Alcalá, en sustitución del anterior, Sr. Serafin Olcoz Yanguas, que dimitio como sabeis en Diciembre, y siendo Diego Alcalá el anterior Vicepresidente legalmente constituido.

Secretario: Lorenzo Ayuda

Tesorero: Francisco San Martin

Otros socios existentes:

Sr. Eduardo Porras, colaborador de la realización del Boletín

Sr. José luis Fornies, encargado del Grupo local de Zaragoza

Sr. Serafin Olcoz Yanguas, ex-presidente

Sr. Rafaél Candau, en representación del Grupo local de Sevilla

Para que la asamblea en primera convocatoria, quedará legalmente constituida tenian que concurrir la mayoría de los Socios, que no concurrierón todos los que deberian, y que pasados treinta minutos, se vuelve ha convocar otra asamblea con los representantes allí reunidos, haciéndolo constar en acta y quedando la Asamblea General Extraordinaria legalmente constituida segun el artículo 11 de los Estatutos del Club.

El Presidente en funciones Sr. Diego Alcalá, convoca el nuevo nombramiento de la Junta Directiva que es aprobada por la mayoría de los allí reunidos y quedando reflejada en acta, queda formada de la siguiente forma:

Presidente: Diego Alcalá

Vicepresidente: Lorenzo Ayuda

Secretario: Eduardo Porras

Tesorería: Junta directiva

Todo esto pasado a una acta, para su inscripción en libro de actas del Gobierno civil de Zaragoza.

Hemos tenido varios problemas con la confección de la revista, una de ellas la elaboración e impresión de la misma ya que con la impresora con que se venía realizando no la podemos seguir usando, pero como veis tenemos otra y ya veis que letra tan limpia saca, lo único que esta limitada a 80 columnas pero no es problema, modificaremos un poco el formato de la revista y quedara mejor, como veis.

Para daros una buena noticia, os diré que tenemos 1100 direcciones de nuevos usuarios del QL, más los 300 que ya éramos, aunque la gente va poco a poco renovando: pero aún hay mejores noticias, la revista QUWORLD del mes de Abril, publico un artículo con la existencia de Qlave (ya era hora), y además hace tiempo el Boletín del club ingles QUANTA publico la dirección y la existencia de Qlave. Con todo esto esperamos formar un magnifico club de usuarios del Ql aquí en España.

Siento que este boletín no se haya llegado mucho antes, la causa fué que estando confeccionando la revista, casi finalizada, hubo un fallo del sistema por culpa de una bajada fuerte de tensión, lo que origino la perdida irreparable del fichero, y por mucho que hemos intentado solucionarlo, el Quill nos daba error de E/S, y aunque intentamos crear un progarama desde basic y luego trasportarlo como un programa exportable, que lo conseguimos, una vez que trabajamos con él se corrompía y nos hacía la vida imposible.

Todo esto y más, ya que también se nos calentaba el Ql. Entre la SuperQuboard 512 k y el QUIMI con el RTC, se ponía el Ql como una parrilla, y se nos colgaba a menudo, creerme que ha sido una total lucha contra el ordenador.

Hemos tenido que volver a confeccionar la revista a última hora, en la que no pudimos recuperar varios artículos, pero que serán publicados el próximo boletín una vez que contactemos con sus creadores.

Sin más, saber que aquí seguimos luchando contra tiempo y mareas, y que este Club sigue a toda marcha, sin duda uno de los mejores Clubs de informatica a nivel nacional. Un saludo a todos.

JUNTA DIRECTIVA

DESPEDIDA

Zaragoza 14 de Abril 1988

Estimados Socios y amigos:

He llegado el momento de mi despedida. En el Club he encontrado muy buenos amigos aunque sólo nos conozcamos por carta o telefono, y no voy a citar nombres para no olvidar a ninguno. Aquellos que me conocen y estiman se darán cuenta por aludidos sin necesidad de nombrarlos.

Quiero agradecer todas las colaboraciones recibidas así como el apoyo desinteresado que se ha dedicado tanto a mi gestión como al club. Sin vuestro apoyo habría sido incapaz de fundar Qlave en Noviembre de de 1985 y de ser Presidente hasta Diciembre de 1988: GRACIAS.

También quiero pedir disculpas por mis negligencias o fallos, en los que directa o indirectamente haya tenido algo que ver. De verdad: LO SIENTO.

No me presento a una reelección porque como dije al dimitir, en la editorial del boletin de Diciembre de 1987, sólo me presentaría si conseguía formar un equipo que constituyese una Junta Directiva, no lo he podido hacer. Ademas me despido de vosotros como lo que soy un socio más (socio fundador y ex-presidente, pero socio al fin y al cabo). Si en algo puedo serviros personalmente, CONTAD conmigo.

El Club, aunque el QL hace tiempo que falleció, tiene posibilidades de seguir adelante y de mejorar, ya que el articulo publicado en la QLANDAL de Abril junto con el fichero de unos 1200 socios Usuarios de QL(obtenido de las garantias de HISSA) a los que se les debe informar de la existencia de Qlave, junto con la labor de difusión y el esfuerzo de colaboración en todos los sentidos, de todos los socios, pueden levantar el club como la espuma.

ADIOS, y como dicen: SALUT y FORÇA AL CANUT

Serafin Olcoz

Socio Fundador de Qlave

CORREO DE LOS SOCIOS

Hoy es 4 de Abril y aun no he recibido el boletin de Marzo, lo cual me hace presagiar que seguimos colaborando poco dentro del club.

¿Que ocurre, señores?, ¿tan dificil es escribir media horita al mes sobre los problemas, las impresiones o los hallazgos que hayamos tenido con nuestro ordenador?. Yo no es que lleve mucho tiempo en el club, pero creo que en general toda pregunta solicitada tiende a ser contestada por unos o por otros, y no me cabe la menor duda de que gente que sabe apreciar un ordenador por lo que es, sin despistarse por la publicidad o el márketing, es muy capaz de aportar algo dentro de un grupo que anela conocer cada vez más y más sobre algo que le interesa.

Pero ademas, partimos de la base de que nuestro ordenador, (mientras el THOR XVI no lo remedie) es una máquina de la que no van a salir nuevas versiones, que seguirá en marcha mientras siga existiendo la gasolina del software y la de gente como nosotros. La del software se mantendrá mientras el pirateo no acabe con ella, y la de los grupos como nosotros mientras no nos sentemos a que nos lo den todo hecho.

Quisiera ahora hacer algunas matizaciones sobre algunas errores y algunos malentendidos sobre algunas colaboraciones mías que han aparecido en QLAVE.

En el número de Septiembre de 1987, en el articulo "Como en Gotica", se describe el orden de acceso a sectores pista a pista y por un despiste, en el texto se interpreta la tabla verticalmente cuando hay que hacerlo horizontalmente, así cuando se relata como se lee/escribe la pista 1, el orden de lectura de los sectores es: sectores 6, 9 y 3 de la cara 0, sectores 6,9 y 3 de la cara 1 y luego los sectores 7,1 y 4 de la cara 0 y 7,1 y 4 de la cara 1 no los 2,5 y 8 que se describen en el texto pues estos, tal y como aparece en la tabla son de la pista 2, perdón por el lapsus.

En el número de Enero-Febrero de 1988 al describir la base de datos de farmacologia, digo que es una pena que en la eleccion de fármacos la base tarde 20 minutos. Cuando me llamó Jose Armendi desde Madrid y me dijo que la base de farmacologia no le atraia al tardar tanto en encontrar fármacos, me di cuenta de que faltaban de escribir muchas cosas en la colaboración de Enero-Febrero. He de decir que en localizar un fármaco o un grupo farmacológico, la base tarda aproximadamente lo que se tarda en pulsar la tecla ENTER.

El lapso de 20 minutos se produce cuando mandamos al programa que nos elija en toda la base un grupo de fármacos que cumplan en principio hasta tres características en función de su cinética, toxicidad e indicaciones. La función fundamental es obtener todos los fármacos que puedan ser susceptibles de ser utilizados en el tratamiento de un enfermo en particular.

De esta base, faltan de definir la opcion de creacion de modelo y mientras llega esta, rogaría a aquellos que la hallan obtenida de la libreria que agregaran al procedimiento 'pan' la linea 'haz cp.grupo$=ma.grupo$' linea que es necesaria para un buen funcionamiento de los procedimientos de modificar o añadir y se me habia escapado tras las ultimas modificaciones hechas al programa al incluir el indice farma_gsg.

Aunque creo que casi todos leereis el Sinclair QL WORLD, quiero destacar unos detalles de lo aparecido en el nomero de Marzo. Primero: El Thor XVI se perfila como muy bueno. Va a ser la primera vez que tengamos un nuevo ordenador con QDOS, nuevo en todos los aspectos pues aunque el Thor 20 era bastante nuevo, la placa base del QL seguia ahi con sus virtudes y defectos.

El nuevo modo de video con 16 colores, la utilización de controladores de periféricos especificos para el 68000,el Motorola 68000 como CPU (y en el zócalo del 68000 según aparece en la foto del nomero de Febrero de Sinclair QL World, hay unos sospechosos especios para un 68010/68012 pues los 6 pines inferiores vacios se corresponden junto con el resto del patillaje con un 68012, es decir un bus de direcciones de 30bits, o sea 1Gb. (un 68020 no tiene la misma distribución de conexiones pin a pin y no podria insertarse en ese zócalo), el nuevo FPC los nuevos buffers de periféricos el mantener todas las puertas de conexión del QL incluso la de expansión del sistema (con la salvedad de los microdrives, que creo que nadie se tirara de los pelos), la posibilidad de trabajar en multitarea sin restricciones de memoria hasta 6 Megabytes, ahora que las memorias están bajando de precio a pasos agigantados, puertos de impresora, de ratón pre instalación de disco duro, Floppies de 3½/2, teclado tipo IBM, caja, fuente de alimentación, y creo que hasta un cable con enchufe.

El precio va a estar bastante bien (aunque un poco caro para los precios del AMIGA o del Atari ST, ostensiblemente más barato que un Macintosh) alrededor de unas 145000 pesetas al cambio directo de libras en su version mas barata. Bien es verdad que no tendremos las prestaciones del sonido o de los modos graficos en modo pixel del AMIGA o incluso del ATARI ST pero el seguiremos teniendo los gráficos a escala tan optimos para aplicaciones tecnicas y cientificas y el sistema operativo multitarea más eficiente de toda la competencia junto con el ordenador mas sociable a la hora de comunicarse con cualquier periferico u ordenador y la red local optimizada y.... bueno que os voy a contar.

Junto con la información sobre el Thor XVI, en un pequeñoa recuadrito aparte venia una nueva oferta de disco duro para el QL hecha por CST, en la cual el precio disminuia unas 300 libras respecto a la ultima oferta de disco duro hecha también por CST. Un disco duro de 20Mb y 500Kb de trasferencia por segundo, por unas 140000 pesetas ya es algo que se deba tener en seria consideración.

Respecto del grupo Local de Valladolid, deciros que actualmente estamos en fase de toma de contacto, ya habiendo tenido una primera reunión preliminar el pasado Miercoles 30 de Marzo en el café España de Valladolid, reunión de la que bueno, partimos como objetivo el localizar el mayor número de usuarios del QL en Valladolid. Actualmente estamos en contacto 7 personas y tras de esta primera reunión vamos a intentar contactar con 2 mas. Esperamos que cuando obtengamos las listas de garantias de MISSA en Valladolid podamos contactar con más gente y poder constituir como operativo nuestro Grupo Local.

Mientras el Thor XVI sigue poniéndose a punto de caramelo, y deseando unas frescas y jugosas colaboraciones por parte del personal me despido hasta otra

Nacho Enrique Cabero

QLAVE(217)

NOVEDADES

¿ Qué es un Transputer ?

Según la prensa, los fabricantes de equipos basados en un 68,000 van a abandonar esa linea, y van a lanzar una linea totalmente incompatible basado en el revolucionario T800-20.

Creo que tengo el deber de contar otra versión de la historia del Transputer, pues la prensa solamente piensa en los intereses de los PCs e intentar que el publico se decida por determinados productos. ATARI está preparando software para el Perihelion (nuevo nombre Abad), y ha organizado un equipo que está desarrollando el sistema operativo Helios.
El Transputer tendrá un T800-20 10 MIPS Transputer, con un 68.000 como procesador I/O. Cuatro Mbytes de RAM, ampliables a 64 Mbytes, y un Mbute de dual-port video RAM . Habrá un DMA SCSI port para hard disk, y tres slots de expansión.

Helios será el primer sistema operativo disponible (es muy similar al QDOS), y podrá ser usado en otros sistemas basados en Transputer. Los lenguajes disponibles en su lanzamiento son : occam, C, Fortran, Pascal, LISP y assembler.

Este sistema consiste en agregar una caja, que contiene el Perihelion, a un ATARI Mega ST.

Leon Heller lleva varios meses trabajando en sus transputer modules para el QL (las últimas noticias son que el interface está terminado), y está esperando el 20 MHz T800 floating-point transputer.

Es casi seguro que los transputer se podrán usar en los PCs, pero serán más lentos o por lo menos eso creo.

Las últimas noticias son que el interface transputer para el QL ha sido terminado a finales de noviembre 87 (su precio alrededor de 120 libras). El QL ampliado al máximo con transputers podría alcanzar una velocidad de 120 MIPS.

EL FINAL DEL STRIP POKER (TALENT)

Lo primero de todo, confesar que no sabía jugar al POKER, y he tenido que preguntar las reglas del juego.
La primera vez, después de cerca de una hora, solamente había conseguido dejar a Denise en bikini.
La penúltima vez, llegue a dejarla en Top Less, y solamente me quedaba ganarle sus últimas 40 libras para terminar , pero eran las 21:15 de un sábado, y me estaba esperando Ramon en la plaza de toros de Fuengirola. Naturalmente, me decidí por lo último.

En mi última partida (quizás la última de todas), la cosa empezó con un plato de patatas fritas y un Sin-Cola. Una vez relajado, ya podía empezar la partida sin preocuparme del tiempo, pues tenía a mi disposición un par de horas largas. La estrategia a seguir era bastante sencilla, cuando tenía buenas cartas apostaba hasta el calcetín, y cuando no tenía nada pasaba ahorrando más que un escocés (aunque me marcaba algunas veces un farol con éxito). Resultados : primero le gané el jersey, segundo al pantalón, tercero el sujetador, y ... ya solamente le quedaban 10 libras. La emoción era desbordante, mi hermano pegaba saltos de la emoción (tiene diez años menos que yo, pero él se sabía las reglas mejor que yo, y era mi libro de consulta).

Salen cartas malas, cambio tres, pero no tengo nada. Apuesto, vuelvo a apostar, de pronto la pantalla se vuelve oscura, y suena una música, yo

gritando "Dame un boli, ya tengo el papel, date prisa. Este boli no escribe dame otro date más prisa ".

El primer mensaje no pude copiarlo, pero ponia algo parecido a " Enhorabuena, eres un jugador profesional ", y el segundo " has necesitado 141 jugadas para desnudarme completamente " (you needed 141 games to strip me completely). Solamente me queda decir que Denise en Bikini está muy sexy. En Top Less tiene un par de ..... que son mejores que las de la Sabrina. Y en desnudo total (en este Strip Poker pasa al final desnuda) tiene un mondongo bastante peludo. Una vez que nos hemos cansado de ver a Denise como su madre la ha traido al mundo, pulsamos una tecla para volver a empezar la partida desde el principio.

Realmente, este articulo no sirve para nada, pero creo que debemos colocar alguna nota de humor, de vez en cuando, para no aburrirnos.

Salvador Merino
Fuengirola (QLave 154).

PREGUNTAS

¿Podriais hacer un sencillo programa en basic que moviera un cursor con el ratón de Investronica?

Para pasar en ABACUS de tener una información en filas a tenerla en columnas tengo que utilizar un fichero de exportacion. ¿Sabeis una forma mas directa?

Este ratón sirve para todos los programas con raton o hay incompatibilidades entre distintos sistemas?

Como es posible que la copia de un programa me funcione perfectamente en el QL y esa misma copia no se cargue en el de un amigo (me ha pasado con tres programas)?

¿Podriais comentarme algo el Atari ST?. Estoy mirando una ampliacion de memoria y una unidad de disco para comprar y por menos dinero me venden este ordenador con teclado mejor, 512 K, unidad de disco salida TV...? animadme un poco.

Los que trabajamos con programas de cálculo nos es muy incomodo el manejo de los numeros. ¿Se puede conectar un modulo de numeros como el del spectrum plus o los cajeros automaticos de las tiendas?

Fernando Martinez
(Pamplona-143)

¿Quién está interesado por conocer o aprender FORTH-83 ?. He hecho esta pregunta porque han pasado muchas cosas desde que escribí mi primer artículo para QLave. De los socios interesados en los primeros meses no sé nada, excepto que la mayoría no son socios porque han cambiado de equipo.

Esta pregunta puede ser contestada desde Zaragoza, pues solamente me interesa saber si existe alguien que lee mis infantiles artículos. Mis artículos pueden ser bastante tontos y poco de fiar, pero se ha publicado en nuestra revista más información sobre el tema que en otra publicación nacional de mayor tirada.

¿ Cómo puedo instalar una segunda unidad de disco de 5.25" a mi equipo ?. Mi unidad de disco Cumana de 3 5" consiste en una fuente de alimentación y una unidad de 3 5" NEC, pero solamente tiene un conector a donde va a parar el cable plano (y otro conector para la alimentación).

La verdad, como la única unidad de disco que le he visto las tripas es la mía, no tengo ni idea de como se instala una segunda unidad. Estoy informado de que socios de Quanta y QLave han instalado una segunda unidad de disco de 5.25" o 3" a su SuperQboard o Trump Card.

El miedo que tenía el año pasado ya no lo tengo, pues si se me avería mi QL e HISOFT se niega a recuperarlo (mi QL tiene garantía postventa de Investrónica pero eso es papel mojado) mi software, salvo alguna pequeña excepción sin importancia, está completamente a salvo en disco de 3.5" (los programas protegidos están desprotegidos). Mi colección de Software es bastante grande, y el software es más caro e importante que el ordenador, y puedo esperar a un CST THOR XVI.

He preparado un QUILL V 2.3 para funcionar en multitarea con programas extraños con un area de trabajo limitada a 64K, que es supersuficiente si se tiene en cuenta las limitaciones de QUILL (en los 4 programa de Psion, todo son sorpresas).

Esto lo he hecho con un programa que vagabundea por ahí, MULTITASK, que modifica la versión V 2.3 a un progama ejecutable con EXEC. Y después he limitado su area de trabajo a 64K con la utilidad GRABBER de la QRAM.

Se me olvidaba comunicaros que mi monitor en color lleva, sumando todas las veces, ya más de 10 meses en HISOFT (en marzo serán 11 meses) y aún no me lo han reparado, pero ¿sabéis si saben esos chapuceros, pues llevo gastado más de lo que vale un monitor en blanco y negro (unas 18,000 ₧). Comprar un monitor para el QL en Málaga es algo muy difícil, pues no te venden ni siquiera un adaptador.

Comprar discos de 3 5" en Málaga entre octubre87-enero88 era algo parecido a buscar una aguja en un pajar (de los microdrives mejor no hablar y me he visto obligado a buscar otros proveedores por correo a muy buen precio he llenado 70 discos de la ganadería de los 1.440 sectores).

Voy a contar toda la verdad de una pequeña idea bautizada con el nombre clave Fénix. La idea consistía en colocar en lugares muy visibles de las universidades de Málaga varias fotocopias de portadas muy seleccionadas de la revista QLave con sus señas (para hacerlo he tenido que pedir algún favor a mis amigos de colegio), y un pequeño anuncio gratuito publicado en la revista "El ordenador Personal" Nº 65 1987 en la sección Clubs (por cierto, tiene una errata de un cero menos).

El objetivo era conseguir nuevos socios para sustituir a los que van a dejar serlo.

Lo único que he sacado claro de este asuntillo, ha sido que a los alumnos de informática de Málaga les gustan los dibujos de chicas guapas, y que la operación ha costado la impresionante cifra de 170 €. (Aunque Serafín me ha informado que posee ya una lista de unas 50 direcciones de usuarios en Málaga .

Sobre la revista Ordenador Personal tengo que decir que es la revista de informática más antigua que se edite, trata a todos los usuarios por igual, es la única revista que sigue publicando programas para nuestro olvidado QL, y también la única que un anuncio de un usuario de QL pidiendo ayuda tiene la garantía casi absoluta que será publicado.

Pero esta revista está corriendo el gran riesgo de desaparecer, pues no es puntual en los kioscos, y su venta ha bajado mucho.

Solamente me queda saludar a todos los socios actuales, y comunicar que en mi BUNKER se están cociendo algunas sorpresas.

Salvador Merino
Fuengirola (QLave-154).

TRUCOS

5. Os falla la membrana del teclado de vuestro QL, no preocuparse, que puede haber solucion mientras pedis una nueva al extranjero.
A mi se me rompió una pista y no salian las teclas p<=0 f5 etc, Claro, al fallar la _ no podia poner "load mdv1_" y no podia trabajar con muchos programas basic. Lo que hice fué cargar con f2 el programa KEYDEFINE. (Por cierto, estoy haciendo un indice de los programas que salen en Qlave y donde salen para mandarlo a la revista y es muy de agradecer que todos salen en mayusculas y son facilmente encontrables).

Este programa permite definir con la tecla alt nuevos comandos funciones etc... Yo lo tenia para en los programas con impresora dar a una tecla y que saliera chr$(, que es muy cómodo.
Pero lo que hice fue con ayuda del listado de las teclas que ya estan definidas y borrando los caracteres que me sobraban,hacer un programa de redefinicion de teclas.k_alt"_",k_alt"{",etc... y de esta forma puede utilizar el QL a excepción de los programas en CM que no eran compatibles con el KEYDEFINE.

Fernando Martinez
PAMPLONA(Qlave-143)

Aqui teneis un plano de la pantalla del QL que permite conjugar todo tipo de 'osize' en las coordenadas correspondientes.Por supuesto no sirve para gráficas.La idea es el reducir el original a un DIN-A3 que es muy manejable y facilmente fotocopiable,cosa esta importante,ya que se gastaran muchas hojas.
Para el uso de ventanas no hay mas que recortar en una fotocopia las ventanas adecuadas y ponerlas encima de una principal.Tambien es muy interesante hacer las fotocopias en papel vegetal transparente o bien crearse sus propias tramas de carácteres ,p.e. cuadricula de 2,0) en papel transparente y superponerlas.
He sido muy facil conseguir la cuadricula de 256x512 puntos,ya que la proporcion que salia no era la correcta,quedaba muy apaisada y no servia.Despues de muchas pruebas he decidido hacer un cuadradito horizontal cada uno y medio vertical con lo que sale una proporcion muy acertada.
A un lado coloco los diferentes caracteres con las coordenadas maximas posibles.Tener en cuenta que en modo 8 no salen con ancho 6 ni 8.
Estan marcadas las ventanas que aparecen al dar F2.
No sabia como enviaros el plano, si ya reducido o bien el original. Como es posible que tengais mejores posibilidades de publicacion, quedaros con el original y haced lo que querais con el. Espero que surjan sugerencias y mejoras a esta herramienta tan util.Yo tenia una igual para el Spectrum y genial

Fernando Martinez
Pamplona (QLave 143)

COLABORACIONES

PRACTICANDO CON EL SUPERFORTH (IV)

En el momento de escribir este articulo, estoy algo desmoralizado por motivos que supongo que todos vosotros conoceis ya, pero aún no he tirado la toalla y mi lema es ¡ gloria o muerte En nuestro caso, la muerte es el olvido, el peor enemigo que tiene el usuario de un OL desde hace dos años..

Esta vez voy a escribir sobre un tema bastante clásico una rutina para calcular el dia de la semana de cualquier fecha.

Las formulas que vamos a emplear son :

Dia Gregoriano N = Int(365.25y) + Int(y/400) - (Int(y/100) + Int(30.59.m-2 , + d+32

y = año m = mes d = dia N= Número de dias
W = dia semana (0 = domingo, 1 = Lunes, 2 = Marces,.....)

Y el programa es :

```
: DIA_GREGORIANO ( Año, Mes, Dia --- UD )
  ROT     ( m,d,y ) ( calculo de año abreviado )
    DUP 100 < IF 1900 +   THEN
  ROT  ( d y,m )
    DUP  3 < IF   12 + SWAP
                  1-   SWAP THEN
  2- 3059 UM* 100 UM/MOD SWAP DROP ( calculando el  mes )
  ROT ( y,U,d )  ( calculando el dia )
  32 + + ( y,U )   0  ( y,UD )  ROT  ( UD, y )
  DUP 100 / NEGATE SWAP     ( UD,N,y )100
  DUP 400 / ROT + S-> ROT  ( UD,D,y )
  36525 UM* 100 M/MOD  ( UD,D,mod,UD )
  ROT DROP D+ D+ ;
: D.A_SEMANA  ( Dia_GREGORIANO --- 0_6 ) 3 0 D+ 7 M/MOD 2DROP ;
```

Autor de la versión original : W.C. Elmore, Am. J Phys. 44 482 (1976)
Adaptado a FORTH-83 por Allen Anway, y adaptado al SUPERFORTH por Salvador Merino con la ayuda de, por qué no decirlo, Gerry Jackson autor del Superforth)

un ejemplo de su uso podria ser:
87 1 7 DIA_GREGORIANO D. Imprimirá
1987 1 7 DIA_GREGORIANO D. Imprimirá
ua .
DIA_SEMANA Imprimirá 4, para jueves.

Esto es un pequeño ejercicio para crear un programa calendario, Si alguien está interesado en terminarlo, a qué está esperando, hagalo. Advierto que tengo un par de posibles soluciones sin terminar.

Junto a este articulo he cedido a la libreria de QLave el primer programa escrito totalmente en Superforth. He enviado el listado fuente (JUEGO_CONVERSION_FTH) y una versión para usar en cualquier QL (JUEGO_CONVERSION). La última versión se carga con EXEC_W, pues he usado la recomendación de Digital Precision para Software comercial, la cual borra todas las headers y con la palabra SLEEP me pone la tarea con prioridad cero. El programa consiste en convertir números de una base a otra hasta conseguir cinco aciertos, y volver a empezar seleccionamos otra conversión en el menu.

El único bug que tiene el programa consiste en que si el usuario introduce una burrada que no se pueda convertir en un número con la base actual el programa devuelve el control al teclado en el Superforth, pues en el Stand-alone, el programa y el QL se iran al hiper-espacio. Esto último no lo he solucionado aún, podria alguien ayudarme (ver dos soluciones en el capitulo VI 1).

Se puede introducir en el Superforth datos (números o letras, desde dentro de un programa de mil y una maneras, pero si alguien está interesado en saber cuál es la mejor y más rápida, no puedo contestar esa pregunta. Pero veamos el siguiente ejemplo:

```
DECIMAL

: TEXT ." INTRODUCIR UN NUMERO REAL " QUERY F# CR F. ;
```

Si ejecutamos el programa, la salida debe ser algo parecido a esto :

```
TEXT
INTRODUCIR UN NUMERO REAL 3.14   ( enter )
3.14
```

El Superforth posee una gran cantidad de palabras con las que se pueden crear cadenas y matrices de cadenas, y poder hacer con ellas lo mismo que en el Superbasic, pero más rápido. Aún no he tenido tiempo de probarlas todas, pero en los próximos meses escribiré algo.

Si esto va despacio, yo no tengo la culpa, porque he estado con el tiempo algo escaso, retrasos ajenos a mi voluntad.

Esta serie está un poquillo adelantada, porque necesito tener tiempo para dedicarme a otros asuntos (P.ej el C, el lenguaje ensamblador, desprotecciones, etc ..). Pero puedo adelantar que voy a pedir en mi nueva renovación de cuota de FIG unos libros muy interesantes.

PRACTICANDO CON EL SUPERFORTH (V)

Si el Sr. Fernando Martinez Martinez no me hubiese enviado la información que le pedí el día 26 de agosto de 1987, este artículo aunque habría existido. La información consistía en unos pocos programas de arquitectura y sus formulas, y el objetivo era hacer una versión en Superforth para observar sus ventajas, si hubiese alguna, frente al Superbasic.

Ha sido una verdadera lástima que yo no haya podido finalizar los programas dentro del tiempo previsto, pero os puedo garantizar que no tengo la culpa. En realidad, pasar esos programas del Basic (ZX 81) al Superforth está chupado, pero el riesgo de error es super elevado. Con esto, me voy a proveer bien de patatas fritas, licor de menta, Coca-cola, Ginebra y Coñac, y Dios salve al Superforth, porque si los programas corren es de pura casualidad primero tengo que terminarlos)

Como esos programas usan mucho las matrices y los números reales (la coma flotante), lo primero que he hecho, ha sido definir unas palabras para usar matrices imitando al Basic.

¡ Preparados, todos atentos, ojo que vienen los ARRAYs !

Voy a definir una palabra para usar matrices de una sola dimensión para números enteros de 16 bits.

```
: MATRIZ     CREATE 1+ 0 DO 0 , LOOP
             DOES> SWAP DUP + + ;
```

Su forma de uso es muy simple,

```
12 ( tamaño matriz ) MATRIZ  FERNANDO (  nombre de la matriz ).
```

Para guardar algo,

```
12  3 FERNANDO !      guardara  12 en la posición 3 de la matriz FERNANDO.
3 FERNANDO @          colocara 12 en lo alto del stack
```

Voy a intentar explicar cómo se ha definido esta palabra.
Mucho ojo porque estais observando una de las ventajas del FORTH. Lo que hace es crear una entrada al diccionario llamada FERNANDO (en nuestro caso), reservar los suficientes bytes y llenarlos con cero (según el tamaño que hemos introducido en el stack antes), y lo que viene después de DOES) son los cálculos que hará el programa para saber cuales son los dos bytes que vamos a usar. La formula usada es : ad + 2 * X

Otra palabra, pero para matrices de dos dimensiones para números de 16 bits :

```
: MATRIZ_2   CREATE DUP 1+ , 1+ SWAP 1+ 0 DO 0 , LOOP
             DOES> DUP @ 3 ROLL * ROT + DUP + 2+ + ;
```

Un ejemplo de su uso :

```
3 3 MATRIZ_2 JUAN
34 0 3 JUAN !            Guardará en la posición (0,3) el número 34.
0 3 JUAN @               Colocará el numero 34 en lo alto del stack (TOS).
```

La formula usada es :

- En la primera linea hemos guardado (A,B) el número B+1 en los primeros dos bytes.
- ad + X * B+1) + y + 2

Otra palabra para usar matrices de una dimensión, pero esta vez para números de 32 bits :

```
: 2MATRIZ  CREATE 1+ 0 DO 0 0 , , LOOP
           DOES> SWAP 4 * + ;
```

Un ejemplo de su uso :

```
3 2MATRIZ JOSE
100200.  3 JOSE 2!        guardará el numero 100 200 en la posición 3 JOSE.

3 JOSE 2@ D               Imprimirá el número 100.200 en la pantalla.
```

La formula usada es : ad + 4 * X

Otra palabra para usar matrices de dos dimensiones, pero para numeros de 32 bits :

```
: 2MATRIZ_2   CREATE  DUP 1+ , 1+ SWAP 1+ * 0 DO 0 0 , , LOOP
              DOES> DUP @ 3 ROLL * ROT + 4 * + 2+ ;
```

Un ejemplo de su uso :

```
5 5 3MATRIZ 2 MALAGA
300300  5 0 MALAGA 2!      guardará el número 300 300 en la posición 5 0 MALAGA
5 0 MALAGA 2@ D.           imprimirá en el número 300 300 en la pantalla
```

La formula usada es : ad + 4 + (X (B+1) + Y) + 2

Esto empieza a volverse un poco reiterativo, pero en fin, otra palabra para usar matrices de una dimension, pero esta vez para números reales :

```
: FPMATRIZ CREATE 1+ 0 DO 0 0 0 , , , LOOP
           DOES> SWAP 6 * + ;
```

Un ejemplo de su uso:

```
10 FPMATRIZ ROLLO
3.14 5 ROLLO F             guardará 3.14 en la posición 5 ROLLO.
5 ROLLO F@ F.              imprimirá en la pantalla 3.14
```

La formula usada es : ad + 6 * X

Otra palabra (¡ pero todavia hay más !) para usar matrices de dos dimensiones, pero esta vez para números reales:

```
: FPMATRIZ_2  CREATE DUP 3+ , 1+ SWAP 1+ * 0 DO 0 0 0 , , , LOOP
              DOES> DUP @ 3 ROLL * ROT + 6 * + 2+ ;
```

Un ejemplo de su uso:

```
5 5 FPMATRIZ_2 ADIOS
1.41 5 0 ADIOS F!          guardará 1.41 en la posición 5 0 ADIOS
5 0 ADIOS F@ F.            imprimirá 1.41 en la pantalla.
```

La formula usada es : ad + 6 * (X * (B+1) + Y) + 2

Naturalmente podemos seguir definiendo matrices de tres o más dimensiones sin son necesarias, pero existe una limitación muy importante, la memoria, y en el Superforth tenemos muy poca (existe un pequeño truco usado en sistemas forth con memoria segmentada, la multitarea).

Si alguien necesita definir una matriz de tres dimensiones, aquí tiene una pequeña pista : X * ((B+1)*(C+1))+((C+1)*Y)+Z

Se pueden realizar las matrices, pero tenemos que definir otras palabras Afortunadamente son casi las mismas, pero más cortas. P.e.:

```
ARRAY CREATE DOES> SWAP DUP + + ;
```

Un ejemplo de su uso:

```
ARRAY  MARIA   12 , 15 , 21000 , 15000 .

0 MARIA @ .          Imprimirá en la pantalla 12
```

Como habréis podido observar, es todo análogo a lo anterior, pero en vez de inicializar la matriz toda llena de ceros, la llenamos con todos los datos que deseemos.

Tengo fama de ser super despistado, y creo que no he olvidado de algo. Por si alguien lo estaba hechando en falta, creo que he olvidado definir alguna matriz para almacenar bytes.

```
BYTE_MATRIZ CREATE ALLOT DOES> + ;

: BYTE_MATRIZ CREATE 1+ 0 DO 0 C, LOOP DOES> + ;
```

Estas dos últimas definiciones son casi idénticas (Por favor, mirar antiguos artículos míos para más información), y su uso análogo a todo lo repetido en este artículo, pero con la diferencia de usar C@ para poner y C! para coger números en el TOB.

Se podría seguir escribiendo del tema indefinidamente, porque se han escrito libros enteros sobre el tema, pero no tengo suficiente tiempo, no soy programador profesional, no he estudiado nunca informática en una academia, soy coleccionista, y si existe una causa para seguir escribiendo gratis para QLave, siempre ha sido ese maravilloso y olvidado QL.

En el Superforth, tenemos unas 30 palabras para crear y usar cadenas de matrices, pero no he tenido tiempo de probarlas todas. Todas las matrices de este artículo son numéricas.

Este artículo ha sido escrito con la idea de hundir la frase escrita al final de la página 33 de QLave Volumen IV N.1 Junio 1987., que dice '... Es recomendable para cálculo (donde se sitúa su origen), pero no para el manejo de cadenas o grandes expresiones matriciales, pues no está previsto para ello...'.

Creo que el señor que escribió ese artículo conoce el Forth menos que yo. En el Superforth tenemos solamente unos 55.000 bytes libres para el usuario, y en

una máquina con unos 857.000 bytes libres es totalmente ridículo se pueden tener varios sistemas simultáneamente en multitarea), y esa es una de las principales razones por las que muchos usuarios Forth de máquinas basadas en un 68 000 principalmente (32 bits) han creado un nuevo FORTH, pero ese Forth es solamente un privilegio de unos pocos privilegiados que siempre nos llevan algunos años de ventaja.

PRACTICANDO CON EL SUPERFORTH (VI)

Hoy es un día lluvioso de enero, estoy sólo, y tengo preparado mi BUNKER con un enorme arsenal, y mis sistemas de animación suspendida, por si acaso. El tema de este capítulo es introducir un número entero desde dentro de un programa, pero comprobando que burrada ha podido meter el usuario.

Primera versión (en Superforth cada uno tiene libertad de hacer lo que desee):

```
10 CONSTANT ENTER

: ENTRADA PAD 20 0 FILL PAD 10 EXPECT 0 0 PAD 1- CONVERT ;

: 'QUE 2DROP CR ." ¿ QUE ? " ;

: INTRODUCIR . " INTRODUCIR UN NUMERO "
  BEGIN ENTRADA  C@ ENTER  <> WHILE 'QUE REPEAT ;
```

Se define la constante ENTER, porque la vamos a necesitar. Este pequeño programa se ejecuta con la palabra INTRODUCIR, y puede ser usado dentro de cualquier programa (en Forth nos gusta la programación modular).

Se imprime el mensaje en pantalla, el usuario introduce un número, si no es un número el programa pregunta ¿ Qué ? y da una nueva oportunidad o oportunidades hasta que introduzca algo que se pueda convertir en un número con la base actual. El número podemos verlo con la palabra ., pero si queremos que sea de 16 bits con un simple DROP se soluciona la papeleta.

La definición de ENTRADA puede tener alguna dificultad PAD deja en el TOS la dirección más baja de un scratchpad area usada para convertir números e imprimirlos) con un tamaño de 84 bytes. FILL llena los primeros 20 bytes con cero.

EXPECT recibe 10 caracteres o menos si ENTER es pulsado. CONVERT convierte la cadena que comienza en la dirección PAD 1 y la suma al número doble 0 0 da como resultado como TOS la dirección del último caracter no convertible (en nuestro caso, si todo sale bien, esa dirección contiene 10, que es ENTER y 2DS un número doble, que es el número que hemos introducido.

Segunda versión:

```
32 CONSTANT ESPACIO

: ENTRADA QUERY 0 0 32 WORD CONVERT ;

: ?QUE 2DROP CR ." ¿ QUE ? " ;

: INTRODUCIR ." INTRODUCIR UN NUMERO "
  BEGIN ENTRADA C@ ESPACIO <> WHILE ?QUE REPEAT ;
```

Es igual que la anterior solamente se diferencia que ahora el último caracter no convertible es un 32, el espacio.
He definido ENTRADA con la palabra QUERY, que simplifica mucho las cosas. WORD deja la dirección de la cadena a convertir delimitandola con un espacio.

Puede que alguien esté buscando como un loco, una palabra para crear números aleatorio, afortunadamente para él ese problema ha sido solucionado hace unos diez años (existen muchas versiones), pero voy a escribir la más vulgar y usada en mi programa Juego Conversión de números.

```
VARIABLE SEED
CR .( Por favor, pulse una tecla ) CR KEY SEED !
: RND SEED @ 259 * 3 + 32767 AND DUP SEED ! 32767 */ ;
```

Si tecleamos 6 RND . se imprimirá un número aleatorio entre 0 y 5. La gracia pega es que el programa siempre imprime la misma secuencia de números que depende de la tecla pulsada al cargar el programa en memoria.

Una palabra para imitar el PAUSE del Superbasic :

```
: PAUSE ." PULSA UNA TECLA PARA CONTINUAR " KEY DROP ;

: PAUSA@ ( N --- ) 0 DO LOOP ;
```

La primera sirve para parar el programa hasta que una tecla sea pulsada. La segunda, según el número que esté en TOS, se perderá más o menos tiempo en ejecutar el bucle DO ... LOOP, que no hace absolutamente nada.

Creo que he demostrado que algunas cosillas en FORTH-83 es extremadamente fácil. En C, tenemos que crear una función algo más larga y lenta

Si alguien se está preguntando cuando terminará esta serie o si continuará solamente puedo decir que si los socios la desean, esta serie puede ser tan larga como la vida de nuestro CLUB.

Salvador Merino.
Fuengirola (Clave-154)

ORDENACION Y BUSQUEDA DE DATOS (I)

En el manejo general de la información, hay dos procesos de vital importancia y que se interrelacionan entre sí: la ordenación y la búsqueda de datos.

¿Para qué se ordenan los datos?, simplemente para poder buscarlos fácilmente, o para poderlos interpretar fácilmente.

Estos procesos son de importancia fundamental en las bases de datos, sobre todo en aquellas que utilizan indices como ARCHIVE y otras.

En el número de marzo de Sinclair QL WORLD, aparece en escena una nueva base de datos para nuestro QL: la "Alpha-base", base que parece tener grandes cualidades sobre todo en cuanto a aprovechamiento del QDOS. En esta base se anuncia como dato importante que utilice el algoritmo (procedimiento en terminología de SuperBASIC) de ordenacion QUICK-SORT en una versión optimizada para el QL. ¿Qué es el QUICKSORT?, ¿Cómo funciona?. Estas y otras cuestiones espero contestarlas en las siguientes líneas.

Ordenación en memoria: Vamos a referirnos ahora a la ordenación de datos en memoria del ordenador, en concreto en forma de matrices; luego nos referiremos a la ordenación de datos en archivos.

En general hay tres metodos de ordenación:

-POR INTERCAMBIO: se van cambiando entre si los elementos desordenados hasta que toda la matriz, esté ordenada.

-POR SELECCION: Vamos colocando todos los elementos de una matriz en otra o en un fichero seleccionando siempre el elemento menor (o el mayor, según queramos ordenar la matriz)

-POR INSERCION: Vamos cogiendo elementos uno a uno de una matriz y los vamos colocando en otra en su posición correcta.

Hay muchos procedimientos de ordenación de cada tipo, pero en general debemos valorar para determinar cual utilizar los siguientes parámetros rapidez del procedimiento en el caso medio, rapidez en el caso mejor y en el peor, si tienen o no un comportamiento natural en el sentido de tardar menos si la matriz está parcialmente ordenada, y si reordena elementos con claves iguales. Todos estos parámetros influyen en la velocidad del procedimiento, sobre todo en matrices de grandes dimensiones (en matrices de pequeñas dimensiones la variación de velocidad de un procedimiento a otro no será casi detectable).

Vamos a ver ahora primero un ejemplo de ordenación de cada tipo antes dicho, y luego dos ordenaciones optimizadas la Shell y la Quicksort.

Ordenación por intercambios: procedimiento de la BURBUJA:

Tal vez ésta sea la ordenación más conocida, la más simple, pero tal vez de las peores. El sistema es hacer comparaciones repetidas, y de precisarse, intercambia el orden de elementos adyacentes de tal forma que los elementos se van moviendo por la matriz del mismo modo que las burbujas de aire ascienden en el agua. Para este y todos los demas ejemplos vamos a utilizar una matriz unidimensional definida como DIM matriz(n) de cualquier dimensión.

```
DEFine PROCedure BURBUJA
FOR a=2 TO a=DIM(matriz,1)
   FOR b=DIM(matriz,1) TO b=a STEP -1
   IF matriz(b-1) > matriz(b)
      t=matriz(b-1)
      matriz(b-1)=matriz(b)
      matriz(b)=t
      END IF
   END FOR b
END FOR a
END DEFine
```

El bucle externo inspecciona la matriz entera, el bucle interno es el que realiza los cambios pertinentes de modo relativo de cada elemento con todos los demás. Con este método siempre se realizan 1/2(n2-n) comparaciones (n-1 veces el bucle externo multiplicado por n/2 veces el bucle interno de ellas), se realizan 0 intercambios en el mejor caso y 3/2(n2-n) para el peor caso (hay 3 intercambios en una ordenación de la burbuja por cada elemento desordenado). Para un caso medio serían 3/4(n2-n). El problema de este tipo de ordenación es el gran número de comparaciones que hay que realizar, factor de n cuadrado.

Ejemplo:

Si queremos ordenar en orden creciente la secuencia de números 7,5,4,6,3,1,2 con el anterior método, los resultados a cada vuelta del bucle exterior serían:

```
7,5,4,6,3,1 2
1,7,5 4,6,3,2     intercambios 15
1,2,7,5,4,6,3     intercambios 15
1 2,3,7,5,4,6     intercambios 12
1,2,3,4,7,5,6     intercambios 6
1,2,3,4,5,7 6     intercambios 3
1,2,3,4,5,6,7     intercambios 3
```

Ordenación por selección:

Para ordenar así, elegimos por ejemplo el elemento con el menor valor y lo intercambiamos por el primero, después, de los restantes se elije el de menor valor y se cambia por el segundo y así sucesivamente hasta llegar al final: en nuestro caso de una matriz(n) sería así:

```
DEFine PROCedure SELECCION
FOR a=1 TO DIM$(matriz,1)-1
c=a
   FOR b=a+1 TO DIM$(matriz,1)
        IF matriz(b)<matriz(c)
           c=b
        END IF
     END FOR b
   IF c<>a
      t=matriz(c)
      matriz(c)=matriz(a)
      matriz(a)=t
      END IF
END FOR a
END DEFine
```

El número de comparaciones en este método casi es el mismo que en el caso de la ordenación por burbuja: (n-1)*((n/2)+1) pero en éste caso el número de intercambios es mucho menor, ya que como mucho habrá n-1 intercambios de elementos cada uno en tres pasos muchos menos intercambios que en el caso de la burbuja, resultando algo más rápido en este sentido. A esto habría que añadir los intercambios en un paso de la variable c dentro del bucle b tiempo que de todos modos no haría que ésta ordenación fuese más lenta que la de la burbuja.

Con nuestro ejemplo:

```
7,5,4,6,3,1,2
1,5,4,6,3,7,2     intercambios 8
1,2,4,6,3,7 5     intercambios 8
1,2,3,6,4,7,5     intercambios 5
1,2,3,4,6,7,5     intercambios 4
1 2 3 4,5,7,6     intercambios 5
1,2,3,4,5 6,7     intercambios 4
```

Lo que supone 20 intercambios menos que con el método de la burbuja (y éste es un caso bastante extremo).

Ordenación por inserción

En esta ordenación, primero ordenamos los dos primeros miembros de la matriz, después insertamos el tercer elemento en su posición correcta respecto a los dos primeros, luego el cuarto se inserta en la lista de tres elementos, el proceso continua hasta que todos los elementos han sido ordenados. Para nuestro caso tendríamos el siguiente procedimiento

```
DEFine PROCedure INSERCION
FOR a=2 TO DIM(matriz,1)
t=matriz(a)
b=a-1
   REPeat ciclo
       IF (b=0) OR (t>matriz(b)):EXIT ciclo
       matriz(b+1)=matriz(b)
       b=b-1
   END REPeat ciclo
matriz(b+1)=t
END FOR a
END DEFine
```

En éste método el número de comparaciones varía de un modo natural de acuerdo al grado de ordenamiento a priori de la matriz, haciendo desde n-1 comparaciones si la lista está ordenada hasta nuestro 1/2(n2+n,-1 en el peor de los casos siendo el total de intercambios 2(n-1) en el mejor de los casos y 1/2(n2+3n-4) en el peor. Siendo en el caso más extremo un algoritmo tan fatídico como los de la burbuja o de selección, aunque en casos medios es ligeramente mejor por su comportamiento "natural". Además tiene una ventaja adicional, si una lista está ordenada de acuerdo a dos claves, después de reordenarla por inserción, permanecerá ordenada por dos claves.

Volvamos a nuestro ejemplo (algo siendo bastante extremo por cierto)

```
7,5,4,6,3,1 2
5,7,4,6,3,1,2     intercambios 1
4,5,7,6,3 1 2     intercambios 2
4,5,6,7,3,1,2     intercambios 1
3,4,5,6,7 1 2     intercambios 4
1,3,4,5,6,7,2     intercambios 5
1 2 3,4 5 6,7     intercambios 5
```

En total de intercambios, supone 16 menos que por sustitución y 36 menos que por la burbuja, y el número de comparaciones en este caso es parecido al de intercambios

Hasta ahora, hemos definido tres métodos generales que en casos extremos son muy malos aunque en casos "normales" son progresivamente mejores
En próximo mes avanzaremos un poco más y describiremos dos métodos de ordenación menos intuitivos pero que son más eficientes en casos extremos y más "naturales" en su comportamiento y por cierto, de los más utilizados en los programas de ordenador (que por algo se les llama ordenadores : Shell y Quicksort.

Nacho Enrique Cabero
Valladolid
(QLAVE 217)

CONJUNTOS, BITS, BASES DE DATOS, OPERACIONES LOGICAS y SuperBASIC

El título podría parecer un poco enrevesado, pero como se verá más tarde es un título muy coherente. A primera vista ya se puede deducir que en este diserto, vamos a hablar de SuperBASIC.

En el conjunto de operadores del SuperBASIC hay unos operadores muy interesantes y que muy rara vez se utilizan (tal vez porque sólo se nombran de pasada en la guía del usuario), pero son los que nos van a permitir lograr los objetivos que aquí me propongo explicar. Son los operadores binarios.

En el SuperBASIC existen cuatro operadores binarios representados por unos símbolos ASCII dobles específicos:

Y lógico inclusivo (AND)	Mayúsculas + 7
O lógica inclusiva (OR)	CTRL + 0
O lógico exclusivo (XOR)	Mayúsculas + acento izquierdo
Negación (NOT)	CTRL + paréntesis cuadrado de cierre

(las traducciones de teclado corresponden al QL español)

Estos operadores trabajan ejecutando las funciones lógicas que se especifican con cada uno de los 16 bit de los numeros enteros en que operen, de hecho ejecutan las operaciones más rápidas y básicas que es capaz de ejecutar la Unidad Aritmética Lógica (ULA) de la CPU MC68008 (Con estas operaciones sencillas, la CPU ejecuta todo su repertorio de instrucciones), señores tenemos aquí a los operadores más rápidos del SuperBASIC (y algunos sin habernos enterado todavía). Estos operadores, como ya se puede intuir, permiten hacer averiguaciones a nivel de bit

Los bit individuales, son ,como harto estamos de oir, las mínimas unidades de información indican 1 o 0, presencia o ausencia... sí o no blanco o negro, . significan en realidad lo que nosotros queramos que signifiquen, sin ninguna restricción Pero como los homos más o menos sapiens tienen ya desde los años de Maricastaña la costumbre de almacenar sus sapiencias con unos garabatos que se llaman numeros y letras y que por mala suerte son más de dos, resulta que para los aparatos que solo saben decir sí o no, como en este caso nuestro querido Sinclair QL, para poder llegar a tener tantas representaciones como garabatos, se agruparon estos bit en grupos de ellos grupos que según el coro de ues y noes que formaran irían identificando cada uno de los garabatos en cuestion. En el mundo Occidental estos caracteres son alfanuméricos formados normalmente por unas 26-30 letras según el país y 10 números más una serie de caracteres especiales que conforman lo que se llama ortografía.

Bueno con un grupo de 6 bit ya tenemos 64 (2 elevado a 6) combinaciones distintas de unos y ceros, con lo que ya podríamos representar esos signos y podríamos entendernos.

De hecho las antiguas maquinas que utilizaban cinta perforada de papel de 7 canales (el central es de arrastre) utilizaban un código de éstos el más común se llamó BCD, pero como las minúsculas no entraban en este código había que agrupar bits en coros de mayor resonancia y para no desequilibrar a los tractores de papel se utilizó un código de 8 bit (cinta de 9 canales) parecido al anterior que se llamó EBCDIC (256 caracteres)

Más tarde cuando empezaron las transmisiones de datos codificados por una sola línea apareció otro código de 7 bits (128 caracteres tambien con minúsculas) más bit de paridad (para comprobar si la trasmisión era correcta. Este código es el ASCII, este código acabo difundiéndose mucho dado que la transmisión masiva de datos por línea provocó que muchos ficheros estuvieran con caracteres en ASCII Con el tiempo apareció una versión del ASCII extendido para englobar otros caracteres exóticos que quisiera englobar el fabricante de ordenadores, normalmente al objeto de poder acoger el mayor número de sensibilidades lingüísticas posible, y éste código ASCII extendido en su versión QL es la que disponemos.

Bueno dejémonos de historias y vamos a los bits.

Todo lo anterior viene a que si con conjuntos de bits hacemos representaciones de elementos distintos, cuantos más bits agrupemos más elementos distintos podremos identificar. Así con grupos de 16 bit que son los que confayman los números enteros del SuperBASIC se pueden representar 65536 elementos distintos que representan a los números del -32768 al 32767 cero incluido. Ahora bien, nada nos impide dar la vuelta a la tortilla y tener que con 65536 cifras distintas podemos identificar la presencia o ausencia de cada uno de los 16 bit que hay en el conjunto en cuestión, bits que normalmente se identifican del 0 al 15 según la potencia de dos que representen (en aritmética sin signo pero que nosotros podemos identificar de otro modo por ejemplo, un fabricante de pinturas que utilice 15 colores básicos (no vamos a trabajar con 16 colores para evitar de momento problemas de desbordamiento en SuperBASIC dado que la aritmética de números enteros del SuperBASIC es con signo))

```
Rojo_bermellón=0                    Rojo_Cereza=1
Naranja=2                                   Salmón=3
Amarillo_real=4                     Amarillo_limón=5
Verde_Nilo=6                                Verde_esmeralda=7
Azúl_Ultramar=8                     Azul_Prusia=9
Azúl_añejo=10                               Violeta_florido=11
Morado_ciruelo=12                           Blanco_de_Plata=13
Negro_azabache=14
```

(es suponemos variables inicializadas de SuperBASIC)

Podría equiparar cada una de esas 32768 cifras con cada una de las combinaciones de colores que se pueden formar con esos colores equiparando la presencia de un color con un uno en un bit determinado (numerados del 0 al 14 según la lista anterior), y la ausencia de color básico con un 0 ... y así identificar cada uno de los colores de un catálogo de pinturas con uno de esos 32768 números (no quiere decir que tenga tanto surtido de colores disponible).

Trabajando así estamos trabajando con conjuntos de colores, si... con los famosos conjuntos de las matemáticas de nuestra infancia en la escuela.

Imaginemos ahora que nuestro fabricante de pinturas recibe un pedido de una serie de pinturas del catálogo y desea saber que colores básicos va a precisar para atender el pedido, con que OKee (esto no tiene nada que ver con la curva jamón serrano sino con la función O lógica inclusive que vimos al principio), todos los números de pedido obtendrá un número que contenga todos los colores que buscaba.

Si no tienes claro lo que hace la función OR puedes mirar los artículos sobre lógica que aparecieron los meses de Julio, Agosto y Septiembre del año 87 en el boletín (la función OR a nivel de bit devuelve un 1 si cualquiera de los dos operandos es un 1). La operación sería una cosa como esta:

```
num común% = num_catal1% AA num_catal2% AA num_catal3% AA num catal4%
```

Con este número común num_común podremos averiguar que colores precisamos así %, el bit 0 de este número es un 1 tendrá el color rojo bermellón en el pedido si el bit 4 es un 1 tendrá amarillo real en el pedido y así con todos los 16 bits cuyo del 0 al 14

Pero num_comun es una cifra decimal. ¿Cómo sacar la lista de los colores que hay en éste número?, Muy sencillo:

Si hemos definido una serie de variables con nombres de colores con los valores del 0 al 14 que antes se dijeron, con un programa SuperBASIC como el siguiente podemos sacar la lista de colores en cuestión:

```
IF num_común% && 2^Rojo_Bermellón: PRINT "Rojo Bermellón"
IF num_comun% && 2^Rojo_Cereza:PRINT "Rojo Cereza"
IF num_común% && 2^Naranja:PRINT "Naranja"
.
.
.
IF num_comun% && 2^Negro:print "Negro"
```

Utilizamos el operador Y lógico inlusivo (AND) pues con él la comparación IF da cierta si en los dos operandos hay un 1 en el bit seleccionado por cada una de las distintas potencias de 2.

Pero hacer 15 comparaciones IF no parece algo muy ortodoxo que digamos, parecería más lógico construir una matriz de nombres de elementos del conjunto de colores y explorar los bit con un bucle for:

Para construir la matriz se podría hacer así

```
DIM SET$(14,25) :REMark 15 cadenas de 25 caracteres
RESTORE
FOR f=0 to 14
READ a$
SET$(f)=a$
perador Y
DATA "Rojo Bermellon","Rojo Cereza","Naranja","Salmón" .... ,"Negro"
```

Luego para averiguar la lista de colores de ese num_comun bastaría hacer:

```
FOR f=0 to 14
IF num_comun% && 2^f :print SET$(f)
END FOR f
```

bucle que es más coherente con una buena programación.

Resulta que en el pedido hay que utilizar el color Azul Prusia el Rojo Cereza entre otros, pero que en éstos casos las existencias en almacen de éstos colores bás cos se están agotando. El fabricante llama a sus proveedores y éstos le dicen que debido a una huelga de los Prusianos no se podria atender el pedido hasta dentro de un mes y del otro lado que debido a un raid de fabr cante de Morado comiona, han desaparecido todas las cerezas disponibles y que el color Rojo Cereza no será posible enviarlo antes de un mes

Nuestro fabr cante se ve entonces en la obligación de avisar a sus representantes de que los colores del catálogo que contengan éstos colores bas cos no podrá servirles durante los próximos 40 días, pero ¿cómo averiguar de su largísima lista de colores (imaginemos que tiene unos 2000 de los 32768 posibles , cuáles contienen los colores 1 y 9 (Rojo cereza y Azul de Prusia), para confeccionar la lista de colores en suspenso?.

Si el fabricante tiene en un fichero fipi_Carta_de_colores todos los números de referenc a de los 2000 colores disponibles, todos seguidos, este fichero ocupará a 16bit por color, 4000 octetos, (quién nos iba a decir que con sólo 4000 octetos podríamos decir 15 cosas distintas de 2000 elementos). Para dar la lista de referencias de color de la carta que no podrá atender el próximo mes, nada tan sencillo como el programa siguiente:

```
OPEN_IN £3, fipi_Carta_de_colores
clave%=2^1 AR 2^9:REMark 1 y 9 són los colores en cuestión
REMark ahora clave% está formado por 14 ceros y 2 unos en las posiciones
REMark de bit 1 y 9
REPeat leer
color%= CODE(INKEY$(£3,-1))*256+ CODE(INKEY$(£3,-1))
REMark esta ültima líneas, los que tuvieran el Toolkit () la podrían
REMark sustituir por GET£3,color% a fin de cuentas se trata de sacar
REMark un número entero de 16 bit del fichero Carta_de_colores que forme
REMark un numero de color del catálogo. Ahora comprobemos fin de fichero
IF EOF,£3):t EXIT leer
REMark ahora elegimos los colores que poseen alguno de los colores base
REMark de clave%
IF color% && clave% :PRINT color%
REMark con éste se imprimen en pantalla los numeros de color buscados
REMark en efecto, con que color% tenga alguno de los colores de la clave
REMark el resultado del AND bit a bit será mayor de 0 y el IF afirmativo
REMark Nada nos impide imprimir de en vez de en la pantalla en una
REMark impresora o en un fichero aparte
END REPeat leer
CLOSE £3
```

(Conste que de quitar todos los REMark el programa se queda en 7 líneas)

Si, en vez de intentar averiguar los colores con ALGUNO de los colores básicos de clave% hubiéramos querido averiguar los colores que tengan TODOS los colores básicos de clave% habríamos tenido que cambiar la línea:

```
IF color% && clave%:PRINT color%
```

por la siguiente línea

```
IF clave%=color%&&clave%:PRINT color%
```

como el lector avispado habría deducido con facilidad.

Llegados a este punto convendría escribir las cuatro reglas de oro sobre las operaciones lógicas a nivel de bit, útiles en cualquier lenguaje de programación que llegue a este nivel. El párrafo procede del libro '68000/68010/68020 Arquitectura y Programación en Ensamblador' de Stan Kelly-Bootle y Bob Fowler del Brupp WHITE, editado por Anaya-Multimedia en España.

"OPERACIONES LOGICAS: RESUMEN
Eligiendo apropiadamente la máscara en operando fuente, se pueden alterar determinados bits o flags en el operando destino. Las reglas son:

```
NOT     Se invierten todos los bits en el destino
AND     0 en la fuente: Pone a 0 el bit seleccionado en el destino
        1 en la fuente: Selecciona los bits que permanecerán inalterados
        en el destino
OR      0 en la fuente: Selecciona los bits que permanecerán inalterados
        en el destino.
        1 en la fuente: Pone a 1 el bit seleccionado en el destino
EOR     0 en la fuente: Selecciona los bits que permanecerán inalterados
        en el destino
        1 en la fuente: Selecciona los bits que serán invertidos
        en el destino."
```

Decir que los conceptos de operando fuente y destino en este párrafo se refiere a la estructura típica del código del 68000 suponiendo que en la fuente hay una máscara de bits y el destino es el operando a manipular por la función lógica. En SuperBASIC sería algo así como:

```
dato% = Máscara% op dato%
```

donde op sería uno de los cuatro operadores binarios && || ^^ ~~ y la máscara sería un número de 16 bit con los 1 y 0 de fuente que sirvan a nuestros propósitos. (lo que hicimos antes para crear la clave% de selección, clave% que no era más que una máscara de bits con los bit de color a elegir puestos a uno).

En construir una máscara en SuperBASIC se puede hacer de dos modos

Si no se tiene el Toolkit II
REMark poner a 1 los bit que queramos ej los bit x,y,z OReando
mascara%= 2^x ÄÑ 2^y ÄÑ 2^z

Si se dispone del Toolkit II, se puede trabajar con números b nar,os directamente, ejemplo para hacer la clave% del programa anterior con los colores roje_cereza (1) y Azul_prusia (9) a 1:

clave%= BIN(0000001000000010)
o presciendiendo de los bit a 0 hasta el bit a 1 de mayor orden (9),
clave%= BIN(1000000010)
(no olvidar que el primer bit es el 15 y el último el 0 según las potencias de 2)

A la hora de construir máscaras en SuperBASIC y según potencias de 2, aunque el formato exponencial 2^x es muy claro, supone forzar una operación innecesaria que de utilizarla muy a menudo podría frenar nuestros programas (aunque en realidad dichas conversiones solo se suelen utilizar una vez al construir máscaras), por si alguno quiere poner las potencias ya efectuadas aquí tiene la tabla de potencias de 2 hasta 2 a la 14:

2^0	1	2^8	256
2^1	2	2^9	512
2^2	4	2^10	1024
2^3	8	2^11	2048
2^4	16	2^12	4096
2^5	32	2^13	8192
2^6	64	2^14	16384
2^7	128	2^15	-32768

Bueno aquí hemos incluido la potencia 2^15 que correspondería al 16avo bit de un número de 16 bits. Hasta ahora no lo habíamos incluido para evitar problemas con la aritmética de signo y facilitar la comprensión, pero en el fondo no hay ningún problema con utilizar también este bit, de hecho ahora que me dispongo a completar las premisas del título de este diserto considero oportuno incluir éste bit para un mayor aprovechamiento de las cualidades de estos operadores (aunque suponga un mayor esfuerzo de comprensión).

De lo que decíamos en el título nos falta hablar de dos cosas en profundidad: los SET y las BASES de DATOS.

¿Que son los SET?: Un set es un conjunto en el sentido matemático.
El termino pertenece al lenguaje PASCAL y es uno de los tipos de datos de

este lenguaje tipo al que corresponden unas operaciones específicas y que no son más que nuestros queridos operadores lógicos binarios. Los SET en PASCAL tienen la ventaja de que los puedes definir con una serie de elementos, y luego es el propio compilador el que se encarga de hacer las máscaras en función de los elementos definidos, dejando que el usuario se olvide completamente del orden de éstos elementos en los bit que forman el set. En el fondo nosotros aquí hemos hecho algo parecido cuando definimos las variables de nombre de color con un valor ordinal de bit asignado hace poco más o menos unas 1500 palabras de disparo.

Más en el fondo desde que empezamos este disparo, no hemos hecho nada más que emular de algún modo todo lo que hace un compilador de PASCAL al definir y manejar SETS. La única diferencia es que los SET que permiten definir los compiladores de PASCAL suelen ser de bastantes más de 16 elementos, aunque encadenando en su manejo un adecuado número de ellos podemos trabajar (con un poco más de código) casi como si lo hiciéramos en PASCAL.

Y digo de 16 elementos que es lo máximo que se puede aprovechar un número entero de SuperBASIC. De ellos el uso de los 15 primeros es tan sencillo como lo visto hasta ahora pero el 16avo bit nos puede dar algún problema por culpa de la aritmética entera con signo que utiliza el SuperBASIC. El 16avo bit (el bit número 15) se utiliza como bit de signo de los números enteros, de tal forma que si éste bit es 1, el número será negativo y su valor decimal corresponderá al número formado por los bit 0 a 14 negado y restado 1, así para obtener un número negativo de un número positivo hacemos un NOT de los bit 0-14, sumamos 1 y ponemos un 1 en el bit 15. Alguno dirá que esto es una complicación de mil demonios, ¿No sería más sencillo utilizar números como los positivos y poner simplemente el bit de signo a 1 sin tanta NOT y tanta suma tonta?, pues no, todo tiene su intención...

La razón de que esto sea así estriba en facilitar las operaciones de suma y resta dentro de la CPU y dentro de la memoria del ordenador veamos algunos ejemplos:

```
-32768+32767 = -1  1000000000000000+0111111111111111 = 1111111111111111
                                                       acarreo 0
-1+4=3             1111111111111111+0000000000000100 = 0000000000000011
                                                       acarreo 1
```

Efectivamente, si ignoramos el acarreo las operaciones con signo se efectuan directamente lo que ocurre es que se cambia el convenio de llamar a los números así los números del 32768 al 65535 que son los que tienen el bit 15 (el 16avo) a 1 se adoptan por convenio que representan los números del -32768 al -1. En el caso de las restas ocurre igual, sólo que los que puede pasar es que un número negativo pase a positivo por pasar a 0 el bit de signo, en el ámbito de una resta normal.

¿Cómo podemos utilizar el bit de signo?, bueno se puede usar para nuestros fines directamente puesto que en el fondo nos interesa el número en cuanto a sus componentes no como tal número, sólo que hay problemas de desborde de uti.. zarlos en el mismo sentido que los otros bit, or ello que para que no haya errores habrá que agregar un signo menos al tratar el último bit, y tratarlo de modo independiente. así por ejemplo en la exploración de un número para buscar la presencia o ausencia de componentes como el hecho anteriormente, suponiendo una matriz SET% de 16 elementos (del 0 al 15) con la descripción ahora sería:

```
FOR f=0 to 14
IF num_comon% && 2^f:PRINT SET%(f)
END FOR f
IF num_comon% && -2^15:PRINT SET%(15)
```

En la construcción de mascaras habrá que tener en cuenta que para el bit 15 habrá que hacer AA -2^15, y de hacerlo con el toolkit 2 habrá que hacer una máscara de 14 bit y de ser el bit 15 : hacer un AA -2^15 ó AA -32768 ó AA -BIN,"1000000000000000"). Cualquier otro valor de -BIN produce error por desbordamiento.

Con ello tenemos sets de 15 elementos fácilmente o de 16 elementos un poco más complicadamente pero en cualquier caso muy efectivos.

Bueno, ya sólo nos queda hablar de las bases de datos

La utilización de SETs es un modo muy potente y compacto de almacenar información. Potente en el sentido de que la comparación de informaciones en un set con otros se hace con operadores lógicos binarios, operadores que son los más rápidos en cualquier microprocesador. Compacto porque su información se restringe a la mínima unidad de información significativa en el mundo informático: el bit.
Para comprender esto imaginemos una base de datos de control de stocks pongamos de un comercio de ropas:
Esta base de datos normalmente constará de un número de referencia o de modelo (que podría ser el que facilite el fabricante o uno propio y con el que se podría acceder por ejemplo a un fichero de proveedores), una descripción del artículo y luego los típicos campos de cantidad de stock por tallas preciso de coste, de venta, iva...

La mayoría de estos campos son numéricos, y no abultan prácticamente nada en la base de datos. En cambio los campos descriptivos, alfanuméricos son muy costosos en cuanto a tamaño para almacenamiento, y además suelen ser los más interesantes de cara a las indagaciones en el fichero. Para estos campos se podría hacer una adjudicación de uno o varios sets (con 3 o 4 a lo sumo valdría) en los que se contuviera toda la información.

Ejemplo en este caso un set que dijera que tipo de prenda es falda jersey, abrigo, pantalón, chaqueta, camisa, blusa. .)incluso podría haber espacio para decir si tiene o no forro, o decir que tipo de forro tiene (si hay algún tipo de forro es que está forrada. Como es lógico para cada una de estas posibilidades se asignaría un bit. Luego los siguientes sets podrían ser comunes a todos los articulos (ejemplo descripción del tejido, lana seda, fibra, ...) otro jaspeado, rayado, pata de gallo, cachemir,) otro para colores , otro para matizaciones específicas del tipo de prenda y que sería dependiente en su significado de los datos del primer set . Creo que para este caso en concreto con estos 4 sets habría de sobra y estos cuatro sets ocuparían !AN BOLO 8 OCTETOS DE FICHERO.

Además en caso de que por ejemplo tuviéramos que hacer una selección para averiguar ¿Cuantos pantalones de lona vaqueros, rojos hay en stock, de qué fabricantes y que tallas?, con crear una máscara de bit con los datos correspondientes a la selección (máscará que se podría crear con cuatro menus, correspondientes a cada set) y hacer las oportunas operaciones lógicas de bit (normalmente intersecciones con la mascara, es decir operaciones AND (&&,), podríamos recorrer rapidísimamente el fichero y obtener los datos pertinentes, sin tener que ejecutar las típicas e inexactas comparaciones de cadenas de caracteres.

Como vemos, podemos trabajar con ficheros de un modo muy rápido. Pero si estar la información tan comprimida y ocupar tan poco espacio, es muy posible que se pudiera albergar toda esta información en memoria con lo que las indagaciones podrían hacerse prácticamente a velocidad del rayo.

Estamos llegando ya al final del disarco, y ya tan solo nos quedan las post datas. Van a ser tres:

1.- El uso de los operadores lógicos binarios es el optimo para hacer averiguaciones en los resultados de la función KEYROW del SuperBASIC, sobre todo cuando se quieren detectar combinaciones de tecla.

2.- Todo lo hasta aquí descrito, para un mejor aprovechamiento ha de hacerse con SuperBASIC compilado, ello es debido a que el manejo de números enteros en modo intérprete es aun más lento que el de los números de coma flotante (curioso ¿no?) puesto que en modo intérprete antes de asignar un número a una variable entera se convierte a coma flotante y luego se trunca a un entero tal y como se explica en el manual del QLiberator al describir al funcionamiento del programa de demostración del compilador.
3.- En ARCHIVE no se puede trabajar con SETS a nivel de bit.

Espero que estas reflexiones sobre los operadores lógicos binarios del fabuloso SuperBASIC sean de utilidad, y que la gente se anime a realizar sus

propias bases de datos a medida, bases de datos que siempre podrán albergar todas las matizaciones que uno quiera, con la potencia que uno quiera y que sirvan para un mayor aprovechamiento de la moneda más valiosa de nuestros días que es el tiempo

Nacho Enrique Cabero
(VALLADOLID(QLaser)

Valencia 15-3-88

Hoy me he decidido a escribir unas líneas para este boletín que hoy por hoy es el único medio de información que tenemos los perseverantes usuarios del QL. Realmente no tengo gran cosa que ofrecer ya que mi experiencia informática es breve.

Comenzaré diciendo que soy médico y por eso mi interés va por esos derroteros, creo que somos bastantes los usuarios del ramo, por lo cual sería muy interesante compartir las experiencias de cada uno tal como ha hecho Nacho Enrique Cabero. Yo por mi parte he enviado a la biblioteca un programa de encuestas alimentarias, sirve para ver la composición de platos dietas o menus y hacer encuestas alimentarias a colegios, guarderías etc. se completa con otro programa que analiza los resultados obtenidos y hace un poco de estadística. El programa está pensado para realizar encuestas en centros o a familias. Puede ser de utilidad a los interesados en temas de nutrición.

Otro de mis intereses son los barcos y en concreto las regatas de cruceros. Envío a la librería un programa que permite averiguar el rating de un barco. No explico que es esto ya que es un poco largo y el programa sólo es útil a las personas introducidas en esta materia.

Tambien tengo una gestión de video club con archivo, pero ya que no está completa la dejo para más adelante.

Respecto al club creo que su funcionamiento es aceptable conforme con la falta de colaboraciones, pero creo que se debe a falta de capacidad no ha falta de interés, al menos es mi caso.
Creo además que se podría mandar a cada uno la lista con direcciones y teléfonos de cada socio, el sistema puede ser por la revista o en papel aparte, pero creo que ayudaría a conocerse.

He pedido a SPEM el kit de QL System2, todavia no lo tengo pero en cuanto lo tenga mandare una lineas de comentario. La idea es buena es una caja donde alo a la placa del QL la interface de disco, los discos, alimentadores, mantiene los microdrives accesibles, y aporta un bus de expansion con 4 5 ranuras donde puedes poner por ejemplo la placa DEPROM que permite tener en rom 3 programas, ademas un teclado tipo PC, ventilador, fuente de alimentacion nueva, etc

Respecto a los consabidos PC's he tenido ocasion de trabajar largo y tendido con ellos Manejabamos el paquete PCFOUR de Psion que es casi igual al 2,3 de Psion para los QL y creo que exactamente igual al 2 38 (no lo se cierto , pude ver y demostrar que aunque en ingles es con mucho lo mas facil de manejar que se puede encontrar sin dejar de ser potente. Yo lo utilizaba para demostrar las posibilidades de un paquete integrado sencillo frente a otros mas consagrados tipo OPEN ACCESS y LOTUS 123

Por cierto que este paquete es mas rapido en un QL expandido que en un PC a 8 Mhz

Estoy completamente de acuerdo con lo que comenta Serafin (perdona la familiaridad pero a fuerza de leer editoriales tuvos ya te considero uno de la familia) respecto al montaje de perifericos. Yo sufri mucho para montar la expansion de memoria (SPEM 640K interna) debido a las poco claras instrucciones y a mi inpericia , y no precisamente en este orden). Hoy soy un experto montador de este periferico, si alguno tiene problemas no dude en llamarme. Para los no iniciados en el ingles se me ha ocurrido un sistema para obtener el juego de comandos del archive ingles y su correspondiente en castellano. Consiste en cargar el catellano, editar un procedimiento que comienza por 10 "todos" y despues continua con todos los comandos y funciones disponibles, incluso las palabras que se usan de union como es "como" .

Cuando esta completo el juego se lista por impresora mejor en hojas sueltas), y se salva como objeto.

Se carga ahora el archive ingles, se carga el procedimiento anterior y se edita. Los comandos estan ahora en ingles. Se sustituyen los 10 "all" del principio por "Rem", y se lista sobre el mismo papel que antes. Si hemos acertado en cuadrar el papel tendremos los comandos ingleses en la izquierda y su traduccion en la derecha No es gran cosa pero el menos clarifica.

Tambien va bien el formato de carta en ingles que se publico, yo lo he usado y al parecer la entienden, pues me mandaron rapidamente lo que les pedia

Bien solo dar animos a todos, tanto a los que colaboran como a los que no, que tener un QL entre el mar de PC's que nos invade, ya es bastante osadia.

Miguel Fraquer
Valencia(Qlave-202

COMENTARIO DE PROGRAMAS

LOS LIBROS SOBRE EL MOTOROLA 68000 EN ESPAÑOL Y PROGRAMAR EN ENSAMBLADOR

Las siguientes líneas son el compendio de mis impresiones, después de haber leído los libros que sobre este microprocesador hay en español. Lo voy a hacer siguiendo un orden no de aparición en el mercado, sino de un posible mejor aprovechamiento de alguien que pudiera acercarse por primera vez al lenguaje ensamblador.

Aparte de los libros, los que quieran programas en ensamblador, como es lógico precisarán de un programa ensamblador. De ellos disponemos de cuatro distintos que sucintamente se les podrían dar los siguientes comentarios:

Assembler Workbench: Es el de Talent. Es un paquete formado por un editor, un ensamblador y un monitor. El editor es rapido, pequeñito y bastante potente, además maneja la memoria del ordenador de un modo dinámico y trabaja con ficheros en RAM, muy recomendable. El ensamblador es sencillo y muy rápido, tiene la ventaja de llevar predefinidos los nombres de TRAP y de rutinas vectoriales así como de las variables del sistema. Se puede trabajar con él de modo línea a línea con el monitor o como ensamblador normal en dos pasos.

El monitor es bastante flexible dada su muy buena interrelación con el ensamblador y a destacar su excelente ayuda a punta de tecla. La única pega del monitor es que trabaja los programas que traza como si fuera el job monitor en sí, no de un modo independiente como el de Computer One, y que no es compatible con QRAM.

Computer One Assembler: Paquete formado por editor, ensamblador y Linker. El ensamblador es muy rapido, rapidísimo diría yo, aunque no tiene ninguna complicación ya que el linker que te permite desarrollar el programa en trozos y luego montarlos, o crear librerías de rutinas en ensamblador. El editor tiene la ventaja de una interrelación con el fichero de errores, como es tradicional en los programas de la casa (es el mismo que el del pascal o el forth) y con su misma tendencia a colgarse al borrar grandes bloques de código.

Se puede completar el paquete con el Computer One Monitor sin duda el mejor monitor que hay para el QL (al menos para mí).

Metacomco Assembler Development Kit: Paquete compuesto de editor, ensamblador y linker. Es el ensamblador clásico del QL, aunque es el más lento de todos, y el que más sbuite, tanto que requiere llamar a "overlays" para poder mantener su código en una máquina de 128K, además el editor asigna la memoria de un modo anómalo y en caso de quitar el editor, con 128K luego no entra el

ensamblador en memoria (el que entra pero no arranca). Si en el verdad que la flexibilidad de las macros con parametros ensamblado condicional variantes de código objeto le hacen ser un paquete muy completo. El linker es sencillo y cumple bien su cometido uniendo bloques preensamblados en su totalidad.

GST Macroassembler: Creo que es el paquete más recomendable dentro del mundo de los ensambladores (aunque yo suelo usar el de Metacomco porque fue el primero que utilicé y al que mas acostumbrado estoy a sus sutilezas). El compilador es en dos pasos rapidísimo y no abulta mucho, además permite macros y ensamblado condicional (aunque sin parametros). El editor permite además tener siempre una copia del último fichero modificado como reserva, y es bastante potente (más que el de Metacomco). Junto a estos tres estados del ensamblador son muy profusos juntamente con los macros de subrutinas que proporciona el linkador son herramientas muy importantes a la hora de depurar programas. El linkador es lo mas fantastico que uno se puede hechar a la cara, con una flexibilidad increible en el manejo de librerias de rutinas y control de todos los parametros posibles del programa mediante un fichero de control.

El conjunto de editor, ensamblador, linkador se puede conseguir también a través del Compilador de C de GST, compilador bastante potente aunque con ciertas limitaciones que ya intentè subsanar en un articulo anterior.

Junto a un programa ensamblador se precisa una guia lo más detallada posible del sistema operativo. En Español sólo hay un libro que cumpla los requisitos. QL Programacion Avanzada de Adrian Dickens editado por AA-MM. Este que algunos en el club se han atrevido a calificar como "la biblia" del QL bueno creo que es indispensable, pese a que es un libro que cada vez que lo leas descubrirás algo nuevo relativo que lo habré leido unas 15 veces!. Es el tipico libro de Circuito cerrado, aunque tiene unos capitulos introductorios relativamente asequibles, a poco que te adentres enseguida te encuentras perdido. Yo recomendaria al que lea este libro por primera vez que aunque se pierda que siga leyendo y lo acabe. Cuando vuelva a empezar a leerlo empezará a encontrar que se va enterando de más y más cosas, y cuando lo acabe que la vuelva a empezar a leer y así cuantas más veces mejor. Un dia descubrirás las TRAP, otro dia las colas encadenadas, otro dia los procedimientos de BASIC, otro dia las colas de teclado ... y así poco a poco a quedar asombrado ante el universo de posibilidades tal vez completamente inexploradas que tiene este ordenador.

Mientras se está leyendo el anterior libro, simultaneamente se deberia de ir enterando uno de la base del ordenador: el Motorola 68008. De él hay una breve descripción en el libro de Dickens pero óptimo seria disponer de otro libro que nos ayudara a seguir los tenderes de la programación en ensamblador.

El libro de Lionell Fleetwood 68000 guia del usuario, editado también por AA-MM, puede ser una buena introducción al lenguaje ensamblador, buena en el sentido que es un libro pequeñito, escrito de un modo hasta cierto punto

divertido y que te lleve de la mano a través de las posibilidades y de las técnicas básicas de programación en ensamblador. Tiene eso sí algunas erratas (típicas de cualquier traducción), pero que el lector avispado enseguida descubrirá.

El libro en un principio es bastante docente describiendo bastante bien el procesador y sus instrucciones básicas, las aplicaciones fundamentales las describe paso a paso pero luego las aplicaciones más elaboradas tan sólo da sugerencias de cómo realizarlas, sin llegar a desarrollarlas, eso sí las orientaciones son bastante claras para quienes hayan evolucionado con él en por los primeros capítulos del libro, al final después de haberlo leído a uno le deja con la mosca en la oreja y uno siente la irresistible necesidad de hacer algún programita con un ensamblador.

El libro de José María Angulo Usategui Microprocesadores de 16 bits editado por Paraninfo, es un libro relativamente barato, con el que podemos acercarnos al 68000 (además de al 8086), desde un punto de vista completamente formal. Se nos describen completamente las instrucciones, modos de direccionamiento encapsulado y patillaje, coprocesadores, controladores de periféricos, buses... Es una visión muy física del procesador con leves incursiones en el mundo de la programación, ahora bien dentro de esta descripción formal no falta absolutamente nada, por estar descritos están descritos hasta los ciclos de bus, la codificación y el tiempo de ejecución de todas y cada una de las instrucciones. En su orientación, es el único libro que existe en español, aunque muchas de sus facetas se recogen también en otros libros aquí descritos.

El último libro aparecido en el mercado es el titulado 68000/68010/68020 Arquitectura y programación en ensamblador de Sian Kelly-Bootle y Bob Fowler del grupo WAITE editado por Anaya Multimedia. Es un libro diferente a los anteriores y aunque es el más caro, de no adquirir ninguno más este es el que yo recomendaría a cualquiera.

Este libro trata una introducción de informática básica muy recomendable para cualquiera te va llevando de la mano y sin ninguna posibilidad de equivocarse por el Mundo del 68000 de un modo coherente y muy claro. De hecho ya diría que que sigue los planteamientos que debieron seguir los diseñadores del 68000 cuando este estuvo en fase de desarrollo. El libro te describe por qué hacer las cosas de un modo y no de otro, te ayuda de un modo muy eficiente en la comprensión de las posibilidades del ensamblador y describe matizaciones sobre ciertas instrucciones que no aparecen en ninguno de los libros anteriores. En particular la descripción del manejo de las instrucciones condicionales y de las instrucciones lógicas es excepcional, instrucciones que en el fondo son las que dan el "razonamiento" a los programas. Otras instrucciones que se tratan de un modo cuidadoso son las referentes a la diferenciación de la aritmética con y sin signo aritmética BCD y las sutilezas de los bits de extensión y de acarreo. Sutilezas que sólo aparecen en este libro.

El libro se completa con una descripción de las nuevas habilidades de las CPU 68010/68012/68020 un poco menos asequibles en el 68010 y de un modo un tanto oscuro y denso en lo referente al 68020 siguiendo en éste último caso el tan temible esquema de circuito cerrado que te obliga a leer una y otra vez las mismas líneas.

Espero que estas breves notas sirvan de utilidad a quienes quieran adentrarse como novatos en el mundo del ensamblador 68000 desde el punto de vista del QL, a quienes me permitiría aconsejar que lean y estudien todo lo que puedan y pongan en practica enseguida aquellas ideas de programación en ensamblador que les vaya sugiriendo la lectura sobre el procesador, a fin de cuentas Tony Tebby debió de aprender a programar el 68000 en alguna parte, probablemente mediante algún que otro libro...

Nacho Enrique Cabero
Valladolid

PROGRAMA : SPEEDSCREEN
EDITOR : CREATIVE CODEWORKS
AUTOR : SIMON N GOODWIN

He recibido el SpeedScreen en disco de 3.5" (23-12-87) y un manual de 12 Paginas.

Lo primero que me ha llamado la atención es que la copia original esta protegida. La protección consiste en formatear un disco de 12/252 sectores, y quizás algo más. Copiar y formatear un disco idéntico al original solamente me ha costado unos 3-4 minutos, pues con utilidades como el Toolbox [] y el Supermedia Manager es excesivamente fácil (Ver todos los datos para hacer un formateo idéntico al original).

La verdad, no entiendo como a estas alturas intentan proteger un programa en disco, pues es la mayor tontería que pueden hacer (lo primero que hace el usuario es intentar copiarlo o desprotegerlo). La verdadera protección se encuentra en un número de serie para saber de dónde ha salido la copia.

Lo primero que tenemos que hacer para poder usar el SpeedScreen es cargar el boot del disco original qué a su vez se encargará de cargar todo lo necesario, una vez en programa en memoria nos encontramos con un bonito menu para que seleccionemos la configuración qué necesitamos (existen 8 configuraciones diferentes), las copias de trabajo, para el consuelo del usuario normal, no están protegidas y corren en cualquier dispositivo.

Existen en realidad dos modos principales : el Turbotext es el más rápido, el Colour Sintesis es el más versatil. Normalmente en la mayoria de los casos se usa el último.

Se puede seleccionar la velocidad del scroll con _scroll 2 (2 puede tomar valores entre 1 y 30), que acelera el scroll al doble de lo normal.
Se suministran 8 juegos de letras. Se pueden cargar escribiendo:

```
start=peek(675)
lbytes "fpl_fantasy_fount",start
_fount £2,start
```

Y se puede reset con _fount £2,0
Con el programa QLUDGE_TASK se puede redefinir los juegos de caracteres para crear uno nuevo.
Se han agregado nuevos tamaños de caracteres :

CSIZE	0,0	0,1	1,0	1,1	2,0	2,1	3,0	3,1
XSTEP	6	6	8	8	12	12	16	16
YSTEP	10	20	10	20	10	20	10	20

Se puede poner el Speedscreen en ON o OFF con el comando _SPEED 1 o _SPEED 0 (en mi versión se ha agregado el _SPEED 2).
El SpeedScreen solamente acelera la velocidad en modo 4.
Tabla de factor velocidad,

MODE	CSIZE	[illegible]				SPEED FACTOR
4	0,0	WHITE	BLACK	0	0	5.1
4	0,0	ALL	BLACK	0	0	4.7
4	0,0	ALL	ALL	0	ALL	3.8
4	0,0	ALL	ALL	-1	ALL	4.0
4	0,0	ALL	ALL	1	ALL	4.3
4	1,0	WHITE	BLACK	0	0	9.9 / 12.6
4	1,0	ALL	BLACK	0	0	8.3 / 10.5
4	1,0	ALL	ALL	0	0	6.8 / 8.4
4	1,0	ALL	ALL	-1	0	7.7 / 9.7
4	1,0	ALL	ALL	1	0	7.7 / 9.7

Mi Experiencia con el QUILL se puede resumir en que casi no se nota la velocidad. Pero la inserción y la sustitución funcionan a la misma velocidad y muchisimo más rápido que antes. Un caso curioso, los caracteres Españoles no hacen juego con los demás, están desencastillados (seguramente los ingleses no han pensado en nosotros). La máxima velocidad se obtiene en tinta blanca ! En el Abacus se nota más la velocidad).

Mi experiencia con el editor del Assembler Workbench es bastante diferente a la anterior. ¡Qué velocidad!, ¡Qué maravilla!... El editor del Workbench era rápido, pero con el SpeedScreen ahora parece que tiene velocidad "Supersónica". El cursor se mueve a una velocidad jamás conocida antes en un QL y el scroll parece una "Estela" por lo rápido que es (Te quedas asombrado al ver un borrón en la pantalla de lo rápido que va).

Parece mentira la cantidad de programas tan buenos que están apareciendo en el mercado para nuestro QL. Se anuncian Bases de datos cientos de veces más rápidas que nuestro Super-lento Archive (gracias a su emulador de MS-DOS). El QL no es lento, y no al revés es incluso mucho más rápido que un PC a 8 MHZ. En realidad, lo que es lento es el Software que disponía el QL en su lanzamiento al mercado.

Todo ello ha sido debido a la no existencia de programadores 68 000, pero en los próximos meses el mercado puede tener un gran revés (El Macintosh, después de derrotar a los PCs en USA, está invadiendo Europa!).

Salvador Merino

PROGRAMA : SUCCESS The Supreme CP/M Code Emulation System
AUTOR : BRIAN WATSON
DISTRIBUIDOR: DIGITAL PRECISION

Creo que me vuelto algo loco, pues he comprado SUCCESS. Antes de comentar el programa se contará una pequeña historia.

CP/M es un sistema operativo para ordenadores de un sólo usuario Control Program for Microprocessor). CP/M lleva en el mercado alrededor de quince años, y durante este tiempo se han escrito miles de programas para este sistema operativo. Esta tremenda colección es la principal ventaja frente al Qdos y otros sistemas operativos. Famosos (o infamosos) programas como Wordstar, Borland's Turbo Pascal, MS Basic, Supercalc, Superwriter, Symphony, Lotus 1-2-3, Dbase, Smart, Perfect Writer, Zorland C, Microfocus Cobol, etc., están todos disponibles en CP/M.

Un emulador es un programa que permite la ejecución exacta en un ordenador de un programa escrito para otro, aceptando idénticos datos y produciendo idénticos resultados. Teoricamente la emulación en un MC 68008 es imposible (existen de CP/M cuatro emuladores en el mercado), porque la verdadera emulación se consigue a partir del MC 68.010. Pero de alguna manera, los programadores veteranos se la han ingeniado para implantar la emulación parcial o total en el MC 68000/8.

La posibilidad de emular en los 68000 por software cualquier CPU, sistema operativo e incluso hardware puede permitir en el futuro a los 68 000 apoderarse del mercado en los próximos años (aunque si leen un pequeño artículo sobre los transputers, observarán que nos hemos quedado un poco anticuados. Pero la emulación es insignificante comparado con otras posibilidades que ofrece la familia.

La diferencia del SUCCESS frente a los otros tres emuladores (si tres, no se sabían, verdad, Hallmark, Sandy y QLsoft) es su velocidad que es el equivalente a un Z-80 a 2 MHz. (Bueno eso es lo que pone la propaganda, en el manual pone 1.0 MHz en un Standard QL, y sobre 1.8 MHz en Eprom o en memoria ampliada... Si comparemos esa velocidad con la velocidad de los primeros ordenadores CP/M a 1 o 2 MHz, podremos observar que esta vez no hemos perdido apenas velocidad. Otra característica es que la emulación es total y sin limitaciones. Todo esto lo he podido comprobar, pero si se encuentra alguna sorpresa prometo contarla en un futuro artículo.

Según Digital Precision, el SUCCESS podría ser usado para emular otra máquina basada en un Z-80 (P.e.: un Zx Spectrum) con alguna programación extra. La versión CP/M usada es la 2.2, que es de dominio público (podemos copiarla y regalarla sin problemas).
Existen dos versiones en el disco suministrado, una para 68.000/8 y otra para 68010/20/30. Existe una ligera incompatibilidad debido, casi seguro, al hecho de que el 68000/8 no está preparado para emular.

Para pasar software CP/M a formato QL, si el formato es conocido, tenemos tres formatos definidos:

- DEFINE AMS-DATA (Si el disco en drive 2 es un Amstrad data Format)
- DEFINE AMS-SYST.DRV (Si es un Amstrad System Format)
- DEFINE BBCZ80 DRV (Si es un BBC Micro Z80 Second-processor disk

El formato del BBC se refiere a discos 5.25" DS 80-Track
El formato del Amstrad se refiere a discos 3" Unico cara 40 track

Por si alguien no se ha enterado aún, el QL ampliado con una SuperQBoard Trump Card puede usar unidades de disco de 3.5", 5.25" y 3". Pudiendose usar una pareja de 3.5" o 3.5"/3", pero no me pregunten como se instala, porque no lo sé (en realidad se instala igual que las de 3.5").

Se suministra con el Success un ensamblador 8080, un desensamblador Z 80, un editor un copiador,... ... ... Está totalmente preparado para programar en código máquina Z 80.

El Success solamente puede manejar programas con una longitud máxima de 64K (esta restricción podría ser eliminada en la próxima versión).
En resumen, creo que se trata de un buen emulador CP/M, y en el manual el usuario toda la información necesaria para introducirse de lleno en el mundo CP/M (la lista de comandos es muy grande) que aunque es un clásico, creo que nuestro QL es mil veces más cómodo y amigable con el usuario.

Conseguir programas CP/M, creo que va a convertirse en un gran ENREDO, pues tendremos que copiar el software del Amstrad CPC, y a la vez, creo que va a estar chupado convencer a un usuario de Amstrad para que se pase al QL/THOR. Por lo menos ya tengo el Basic de Microsoft, un linker y un compilador de Basic (este basic no va a usar Mr Sugar, pues si escribo en basic, escribiré en Superbasic), y además también tengo otro emulador de CP/M, que es el cuarto y viene de la mano de QLSOFT. No sé de dónde porras ha salido, pero llegó a mis manos antes que el SUCCESS (los dioses son generosos).
Voy a dar una pequeña idea a los del G.L. de Sevilla, pues según creo, son los que están más especializados en hardware. ¿Qué pasaría si en vez de colocar una Eprom un cartucho conteniendo una Eprom, colocamos una pastillas RAM de 16K ?. Si esto fuese posible, os he dado una pequeña idea Super barata para no tener que estar cambiando siempre de cartucho ROM cada vez que se use un programa en cartucho.
Con este sistema se puede tener una copia en disco de la eprom, cargarla en la misma dirección y después ejecutarla con un CALL. Resumiendo, esta idea tiene dos usos piratear programas sin gastar mucho dinero, y tener 16 K extra para nuestro propio uso.

No sé si es posible, pero se puede hacer, porque existen algunos productos en el mercado que ya lo hacen (la idea la he sacado del SUCCESS, pues en el boot se comprueba si existe un cartucho RAM (en lugar de uno ROM) y si existe carga el emulador de Z 80 en esa dirección.

Salvador Merino
Fuengirola(QLave-154).

PROGRAMA	:	DIGITAL C
AUTOR	:	G. JACKSON
DISTRIBUIDOR	:	DIGITAL PRECISION

Aunque he recibido el programa, he tenido la mala suerte aún habiendo pedido el programa en disco 3.5" 1.440 sectores, de recibirlo en MDV.¿Qué ha pasado ?, pues que va a pasar, pues lo mismo de siempre uno de los ficheros, el CG (code generator/linker), lo he recibido dañado. Solución devolverlo y esperar otro

mas. Estoy harto de mi mala suerte con los programas comerciales en MSX, pues hasta ahora solamente he recibido bien a la primera el QLisp y el Assembler Workbench y si hacemos calculos con la cantidad de programas que he comprado observo que esto dos debió ser una casualidad.

Ya sabe s que no tengo el programa completo, pues me falta el fichero más importante. Pero voy a intentar comentar o advertir de algunas cosas de esta versión. Naturalmente, antes voy a contar una historia de las mias.

El C es el lenguaje más moderno y mejor que existe para un micro. Permite al programador un nivel de control sobre la máquina solamente disponible en código máquina, pero con una fracción de esfuerzo en programación. Si lo comparamos con otros lenguajes, el BCPL (el abuelo del C) está anticuado, ALGOL pendenaje, APL incomprensible, PASCAL austero, FORTRAN rigido, LISP inflexible, COBOL absurdo y SUPERBASIC lento (excepto con Turbo y la consiguiente QLIBERATOR). En esta lista nos hemos olvidado del FORTH. Este último podemos definirlo como olvidado o desconocido. Normalmente en FIG creemos que el FORTH es muy superior a C, y por ello voy a seguir contando la historia actual.

En la última FORTH conferencia se ha tomado en serio la agonia del Forth frente a C, pues este último en los últimos años ha robado muchos usuarios al Forth que ahora programan en C (o en los dos).

Al final todos contentos, se ha votado por mayoria por un nuevo Forth de 32 bits de direcciones, y creemos que el Forth continuará debido a la gran cantidad de usuarios que existen todavia (existe una precisión muy grande para mantener el Forth actual a 16 bit de direcciones por parte de los PCs).

Además la firma Palo Alto Shipping Company, especializada en software para 68.000 (a esta firma cuando le preguntan por qué no escribe programas para PC responde, porque es un mercado muy dificil y demasiado competitivo), ofrece el MACH 2, un sistema FORTH-83 multitarea capaz de linker programas escritos en Forth y C que según ellos ayudará a convivir juntos a ambos tipos de usuarios. Existen versiones para Apple Macintosh, Atari ST y Amiga (aún no terminada).

Supongo que ya alguno estará pensando "valiente rollo se está marcando el tio". Bueno, en realidad no voy a comentar totalmente el programa (porque no lo tengo), pero sí voy a comentar superficialmente con lo que veo en el manual.

Lo primero de todo, decir que R. Jackson es el autor del Superforth (la historia tiene gancho). El Digital C es una implementación del compilador Small-C originalmente escrito para correr en sistemas 8080/Z80, y ha sido modificado para correr en un QL/THOR. El Digital Parser es solamente un entero parser (solamente números enteros) cogiendo un subset del lenguaje C completo.

Una librería es suministrada para manejar aritmética de coma flotantes las salidas / entradas del QL (el Small-C solamente maneja números enteros. Los programas objeto escritos con el Digital C actual están limitados a 64K (incluyendo Kint me dataspace), una nueva versión (será producida si existe una demanda)podría no tener limitación de tamaño.

Con las librerías (funciones C standard, funciones QDOS y librería coma flotante que acompañan al compilador se hacer casi de todo (son bastante completas y fáciles de usar).

En los manuales he observado algunas cosillas que no me han gustado mucho. Por ejemplo, parece que solamente usa números enteros de 16 bits en un QL eso es ridículo), otras cosillas que tengo que probar antes de seguir tirando huevos a este compilador. Creo que Small-C debe su nombre a todas esas cosillas. No sé si será tan rápido como dicen, pero sencillo sí que es

No puedo compararlo con el C de metacomco, porque no lo tengo, pero estoy enterado que compilando es tan rápido como una tortuga Bolichera El QC (GST) después [redacted] todas las cosillas que le ha agregado mi amigo Nacho Cabero (según él, he tenido el privilegio de ser el primero o uno de los primeros en disfrutarlas), cree que ese compilador no es tan malo como muchos piensan, un juego bastante conocido, el Strip Poker está escrito con el QC. Si os estais preguntando cómo nos hemos enterado, simplemente desprotegiendolo (autor IEC Soft).

Naturalmente, se podría contar la historia de todas estas desprotecciones (no la cuento, porque no soy el autor), porque ayudaría a muchos socios a desprenderse de sus MDVs y usar sin piedad sus unidades de disco.

Otra cosilla que no encuentro en el manual del Digital C es cómo introducir líneas en assembler dentro del programa. En el QC (GST) es fácil y está especializado en eso. Y en el C de Metacomco, aunque se puede hacer, es un punto bastante oscuro en el manual.

En el futuro os contaré si vale o no la pena comprar el Digital C. Aunque estoy pensando que la versión actual no tiene ningún futuro. Así que tendremos que esperar otra versión sin los defectos y limitaciones de la actual.

Salvador Merino
Fuengirola Qlave-154

PROGRAMA : ALIEN HIJACK
EDITOR CHI SOFT

La historia comienza en el año 2.003 y usted está trabajando en un barco como técnico. Un día el barco es atacado por una nave extraterrestre, y usted debe recuperar de nuevo el control del barco (aunque más bien parece un enorme castillo o mansión).

Lo primero que tiene que hacer es encontrar tres códigos para tomar el control del puente, que están repartidos aleatoriamente en el barco. Dispone de un láser, puede coger y soltar objetos en otro lugar, correr, saltar, etc.

Tiene que preocuparse por el nivel de energía y munición del láser, que se van recuperando al tocar algunos objetos pequeños repartidos por el mapa del juego.

Hay ocho llaves y cada una abre algunas puertas. Algunos monstruos necesitan ocho disparos para morir.

El juego tiene 192 pantallas a todo color en tres dimensiones. Ya solamente he visto unas 10, pues estoy algo oxidado.

Y aliens diferentes hay muchos (uno muy curioso es una percha de sombreros que se mueve y se para como si estuviese encantada), el diseño de las pantallas es muy detallado. El juego es muy rápido.

El cambio de pantalla es rapido, mas rápida que un Spectrum. Esto no se puede contar, tenéis que verlo para creerlo. El juego hace bastante ruido, pero con el Qtalk se supone que debe ser espectacular.

En resumen, es un juego en la linea del Night Lore y el Fairlight. Y si hubiesen aparecido juegos como éste hace dos años, el QL podria haber sustituido al Spectrum. Los graficos de este juego ponen al QL, a pesar de sus pocos colores, en una buena posición para competir con un ATARI ST o un Commodore Amiga, que han sido especialmente diseñados para juegos.

Salvador Merino
Fuengirola (QLave-184)

Programa: MEGA TOOLBOX
Edita: COMPWARE

Al ver por primera vez este paquete se puede pensar que se trata de una repetición más de los diversos toolkits que ya existen en el mercado, sin embargo no es así. En primer lugar su tamaño ya lo hace algo desacostumbrado ocupa 25 K aproximadamente y nos provee de 170 nuevos comandos

Se trata de un entorno de trabajo que proporciona una orientación distinta a la seguida por los toolkits de Tony Tebby, centrándose fundamentalmente en el trabajo gráfico de pantalla y la animación. En los nuevos comandos podemos distinguir dos tipos básicos por su manera de actuar, uno de ellos es el tradicional, por el cual se consigue la ejecución simple de un hecho, el otro aprovecha la capacidad de multitarea del QL para poner en marcha jobs que se ocupan de realizar las mencionadas actividades gráficas

Veamos ahora estos nuevos comandos por el tipo de tarea que realizan:

- Mejoras en la entrada de datos por el teclado:

Es posible recuperar la última línea de comando y por combinación de las teclas shift + ctrl + cursores, movernos a través de la línea, borrar palabras... Alteración de características como la velocidad de autorepetición, carácter de cambio de ventana... Se pueden redefinir combinaciones de teclas...

- Gestión de memoria:

Permite obtener la cantidad de memoria libre.

Reservar zonas de memoria del área común y liberarlas.
Manipular rápidamente la memoria copiando el contenido de una zona de memoria en otra zona, se puede intercambiar los contenidos de dos zonas determinadas y rellenarlas con un determinado valor.

Almacenar cadenas en memoria bien en formato QDOS (con el número de caracteres de la cadena en una palabra previa) o en formato normal. Igualmente se pueden guardar valores en una zona preparada por MEGA-TOOLBOX de manera que se pueda acceder a ella por diferentes jobs, esta zona es gestionada por medio de semáforos que impedirán que dos jobs accedan al mismo tiempo a esta zona.

Es posible además buscar una determinada cadena por la memoria y obtener el contenido de la memoria en hexadecimal y ASCII.

- Gestión de ficheros:

Obtiene todo tipo de datos sobre los ficheros como su longitud, espacio de datos y directorio extendido con ellos

Permite acceso directo a los ficheros al poder posicionar su puntero y extraer y guardar datos en ellos. Realiza formateado especial indicando el numero de veces que se debe formatear el medio y el número de sectores buenos.

Crea un job que se encarga de copiar el contenido de un medio a otro de manera que el Superbasic continúe su ejecucion normal, algo semejante al spooler del toolkit II.

- Gestión de jobs

Interroga al QDOS por la lista de jobs en curso, su dirección base, identificador, estatus ... . Ejecuta, suspende, elimina, pone en situación de espera a los jobs actualmente en curso.

Tiene una serie de jobs especiales como un reloj, una alarma, permite ejecutar melodías introduciéndole las notas como una cadena alfanumérica, un job que se encarga de ejecutar repetidamente llamadas a código maquina definido por el usuario.

- Gestión de canales:

Permite la localización del ID del canal así como la dirección del bloque de control del canal. Activa pipes como input desde el Basic. Conecta unos canales del Basic en lugar de otros activándoles para input o output y redireccionándolos

- Gestión de ventanas y gráficos:

Como ya he dicho es la parte mas innovadora del paquete y entre sus posibilidades se encuentran las de gestión del cursor de la ventana, selección de distintos alfabetos, en la demo que he podido grabar aparecen 13 nuevos alfabetos ya definidos, algunos de los cuales son realmente buenos.

Activar mediante un solo comando tinta, papel, borde, cursor y alfabeto para una ventana. Obtiene relaciones de posición y amplitud en una ventana tanto en coordenadas como en pixels.

Es posible salvar y cargar una pantalla tanto en una forma normal como comprimida, expandirla luego o salvarla comprimida y cargarla expandiendola. La pantalla se puede guardar en memoria en la zona común y una vez así tenemos una serie de comandos para manipular su posterior paso a pantalla. Así, podemos mostrarla con una máscara sobre ella, intercambiarla con la pantalla actual, podemos aumentarla o reducirla un factor determinado o recolorearla

Se han habilitado versiones de PAN y SCROLL en que la parte que desaparece por un lado se recupera por el otro

Además de todas estas opciones tenemos comandos que habilitan un IOT que producen efectos de animación, así recolorean constantemente una pantalla, alternan colores del borde y flash, recolorean y muestran imágenes almacenadas, realizan pan y scroll continuados...

En lo que se refiere al tratamiento de las cadenas de texto, se les puede dar una tercera dimensión por medio de un parámetro de profundidad. Existe un JOB que admite como parámetro una cadena en la que se incluye el texto mezclado con códigos de control sobre colores, flash, tab, limpieza de ventanas, recoloreados cambio de alfabetos... también se puede escribir un mensaje en una amplitud determinada independientemente de su escala en x e y

Se pueden mostrar imágenes almacenadas con un determinado tiempo de espera entre cada display. Dibuja círculos y elipses rellenos, rectángulos y rellena áreas irregulares Es posible acceder a la segunda zona de pantalla que como sabemos está cubierta por las variables del sistema operativo.

Por fin existen una serie de comandos variados que emulan a algunos comandos del ensamblador del 68000 como por ejemplo testeo de bits, rotación de ellos intercambio de argumentos.. Transforma valores hexadecimales y binarios a decimal y viceversa, obtiene factoriales...

Se incluyen una serie de comandos de tratamiento de cadenas como buscar una subcadena en otra, concatena argumentos, invierte una cadena, Print using.

Por fin y para acabar unas funciones que permiten ejecutar un trap desde el basic, localizar la base de las variables del sistema, las variables del basic, contenido de un registro del micro tras una llamada trap y la dirección base de un buffer permanente de 1k que se usa para intercambiar información entre el Basic u otros programas

Isidro Asín (Qlave-fund)

Programa: QL EXPERT
Editar: COMPWARE

Este programa es una 'concha' (shell) para diseño de Sistemas Expertos, es decir que nos provee de las herramientas necesarias para diseñar nuestros propios sistemas expertos.
El paquete suministrado por Compware consta de un cartucho que consta de los siguientes ficheros:
- Boot programa que carga QL Expert
- Expert El programa en sí, que está escrito en Superbasic y que ha sido Turbo-compilado
- Expert_bin Consiste en una serie de comandos extras usados por el programa principal.
- Expert_hob Es un fichero de ayuda construido a modo de los de Psion.
- Expert_dat Fichero de datos variables dependiendo de la instalación del programa
- Install Nos ayuda a variar parametros como los medios(flp... , etc..
- Backup Programa para hacer copias
- Car_exp
- Human_exp
- Quican_exp Son tres ficheros de ejemplo sobre la construcción de sistemas expertos mediante este paquete.
- Readme_doc Erratas y variaciones del manual.

Además se suministra un completo manual con el cual podremos resolver cualquier duda que se nos plantee respecto al funcionamiento del programa en sí.
Una vez cargados los procedimientos residentes el programa es ejecutable por medio de exec o exec_w y obtendremos la pantalla principal que nos da la bienvenida al sistema por medio de F1, F2, F3, F4, llamaremos respectivamente al fichero de ayuda, ejecutaremos el sistema experto actualmente en memoria, editaremos un menú de comandos, reevaluaremos la última pregunta hecha al sistema.

La fuente de la que beben los sistemas expertos, en general, es su capacidad de razonamiento informal, fundada en una base de datos de conocimientos que ha sido introducida previamente por expertos humanos. La forma de materializarse estos conocimientos es por medio de contenares de Reglas empíricas, fundamentalmente de naturaleza heurística. Las Reglas restringen la búsqueda guiando al programa hacia las soluciones mas probables. Así tenemos por un lado un conjunto de axiomas que configuran el sistema experto en general introducidos por el programador y por otro una serie de datos particulares que serán introducidos por el usuario final en función de las preguntas que le haga el sistema con vistas a la obtención de la valoración o resolución del problema planteado

ompware ha estructurado su Shell entorno a las Reglas , Rules antes aludidas, que pueden ser de tres tipos diferentes:

Reglas de definición, que describen cómo se van a evaluar las diferentes alternativas que nos llevaran al resultado final.

Reglas de input, que permiten al usuario introducir la información adicional al sistema para evaluar los casos particulares.

Reglas de output, que se usan para presentar los resultados obtenidos en el evaluación.

Las reglas de definición pueden ser:

- de carácter ordinario, que tienen el formato siguiente

```
RULE1 num
        DEFINE          nombre
        ( LABEL )       cadena
        ( FACTOR )      num, real
        ( NOT )         nombre1, nombre2...
        ( OR )
        ( NOT )         nombre1, nombre2 .
```

- de carácter selectivo

```
RULE1 num
        DEFINE          nombre
        ( LABEL )       cadena
        ( CONDITION )   condición
        FIRST OF        lista de nombres   ó
        MAX OF              ""             ó
        MIN OF              ""             ó
        ALL OF              ""             ó
```

- funcionales

```
RULE1 num
        DEFINE          nombre
        ( LABEL )       cadena
        ( IF            comparacion
        RETURN          expresión aritmética
        RETURN          expresión aritmética
```

Así pues podemos ver que tendremos nuestro sistema configurado por centenares de este tipo de reglas que nos permiten "definir" determinados nombres, que no son sino una cadena cualquiera, nombres que podrán ser usados en otras Reglas como parte de ellas como sucede en las reglas de definición ordinarias con

nombre1 nombre2 .. o con lo que se llamado lista de nombres en las reglas de definición selectiva. Esta manera de 'programar' el sistema pudiendo usar como nombres cualesquiera cadenas de texto que indiquen claramente lo que cada regla es a hacendo dotan al entorno de una gran claridad de lectura y dotan al sistema experto de otra de sus características fundamentales, es capaz de explicar todas las inferencias realizadas en términos fácilmente comprensibles para el usuario del sistema.

La introducción de estas reglas se realiza por medio de editor de pantalla que se comporta como si de un interprete de Basic se tratara ya que cada frase que introducimos tras pulsar ENTER) es analizada sintácticamente y pasan a mayúsculas las palabras clave que contenga la misma. Existe un menú de opciones que nos ayudan a redactar las Reglas. Es posible seleccionar AUTO que va numerando las Reglas de forma similar a las sentencias de SuperBasic, renumerar las reglas ya escritas, borrar un grupo de ellas, listarlas y buscar una determinada cadena. Las Reglas así escritas se pueden salvar, cargar e incluso mezclar con otro fichero ya existente para formar uno nuevo mayor.

Las reglas de definición pueden estar ponderadas por un factor numérico a la hora de su valoración y existe una gran variedad de funciones a usar en las expresiones aritméticas de las reglas de definición funcional como ABS, BINOM, EXP, FACT, INT, LN, LOG, MAX, MIN, PI, ROUND, SQRT (ESP, que dotan de una gran potencia a este tipo de estructuras al poder realizar valoraciones realmente completas con los datos incluidos en el sistema. Es posible además construir las reglas de manera que nos permitan evaluar las afirmaciones por medio de distintos caminos de manera que no nos quedemos sin una valoración porque el usuario no haya respondido a alguna de las preguntas formuladas. Ante una Regla de input es posible dar unas respuestas 'especiales', así le podemos preguntar 'why' es decir, por qué se nos hace esa pregunta en particular o activar el modo trace, en cuyo caso se van imprimiendo los valores obtenidos en la evaluación de las Reglas sucesivas conforme se vayan obteniendo.

En conjunto pues se nos da una serie de poderosas herramientas para crear un sistema experto a nuestra medida, el hacerlo forma parte de nuestra paciencia para introducir los centenares de Reglas que ha de tener un sistema para ser de verdadera utilidad y capaz de resolver problemas difíciles.

Isidro Asín (Quevefundador)

COMENTARIO DE LIBROS

LIBRO :	LENGUAJE C BIBLIOTECA DE FUNCIONES
AUTOR :	Kris Jamsa
TITULO ORIGINAL :	THE C LIBRARY
EDITORIAL :	OSBORNE/McGraw-Hill
PVP :	2.745 ₧

Lenguaje C Biblioteca de funciones es simplemente un libro de 288 páginas llenas de programas escritos en C totalmente explicados.

Su contenido : una introducción, una revisión del lenguaje, constantes y macros, manipulación de cadenas, punteros, rutinas de entrada/salida, rutinas de manipulación de arrays, recursion, rutinas de ordenación, funciones trigonométricas y conversión de caracteres, programas de manipulación de ficheros, programación del 'pipe' y los códigos ASCII.

Si alguien está interesado en introducirse en el C, este libro no es solamente altamente recomendable, sino imprencindible.

El libro supone que el lector es un programador C avanzado. Es totalmente necesario empezar con un libro como Programación en C. Introducción y conceptos avanzados. Este último lo he terminado de leer y puedo decir que es muy bueno. Pocos libros existen que cumplan su objetivo, enseñar.

Aunque no he podido empezar a leer el libro (lo he terminado y aún pienso lo mismo), lo he ojeado y puedo decir que los algoritmos están suficiente bien explicados para poder adaptarlos a otros lenguajes.

Salvador Merino
Fuengirola (QLave-184

LIBRO :	LISP El lenguaje de la inteligencia Artificial.
AUTOR :	A.A. Berk
TITULO ORIGINAL :	LISP. THE LANGUAGE OF ARTIFICIAL INTELLIGENCE
EDITORIAL :	ANAYA MULTIMEDIA
PVP :	1.950 ₧

El Lisp es el lenguaje de la nueva revolución informática causada por las técnicas de Inteligencia Artificial y los proyectos de ordenadores de quinta generación.

La gran expansión de la Inteligencia Artificial está disminuyendo la distancia entre realidad y ciencia ficción y pronto se producirá una generación de máquinas capaces de comunicarse mediante el habla y dotadas de inteligencia creativa. La importancia del LISP radica en su capacidad para realizar y generar lenguaje natural, que es una característica imprescindible en cualquier sistema informático inteligente.

El Lisp es a la vez fácil de aprender y suficiente flexible y potente como para expresar estructuras lógicas y matemáticas complejas (soy miembro de una secta que creemos que el Lisp es inflexible).
Mediante LISP, El lenguaje de la inteligencia artificial aprenderá los fundamentos, estructuras y técnicas más importantes de programación en LISP, que le permitirán dar sus primeros pasos en este nuevo lenguaje y comprender la proyección real de la IA.

A lo largo del libro se explican en detalle, mediante numerosos ejemplos, los conceptos de recursión, estructuras de datos, proceso de listas y las técnicas y estructuras de programación que hacen del LISP el lenguaje ideal para el desarrollo de aplicaciones inteligentes.

El último capítulo contiene un ejemplo de aplicación orientado hacia la IA, que muestra un caso de proceso de aprendizaje.
Hasta ahora he usado el autocomentario del propio libro, porque está mejor hecho que los míos (hacer comentarios de libros no es mi fuerte, lo mío siempre ha sido las matemáticas).

El Lisp tiene un campo de aplicaciones limitado, pero su fuerte es el manejo de grandes listas de datos, modificarlas y ampliarlas con lógica. Se pueden crear bases de datos que decidan, amplíen, y busquen los datos por cuenta del programa y según una serie de preguntas .

Este libro en cuestión no creo que sea capaz de enseñar a programar LISP si el usuario no practica la técnica de teclear y ver que pasa. En fin es uno de los pocos que hay en Castellano.

Salvador Merino
Fuenginola (QLave-154)

CURSO DE SuperBasic

INTRODUCCION

Son muchas las socias del Club que han pedido un curso de SuperBasic. Nosotros Diego Alcalá y Lorenzo Ayuda, vamos a tratar de organizarlo, aunque sólo somos aficionados como vosotros.

Este curso le interesa a todo aquel que no sabe SuperBasic o que aun conociéndolo no sepa sacarle todo el provecho posible. Para ello vamos a daros unas reglas que aunque son muy rígidas nos permiten optimizar los programas.

El curso lo hemos dividido inicialmente en siete capítulos, en los que podemos variar el contenido a petición vuestra y que en principio están organizados de la siguiente forma:

1-> INTRODUCCION
2-> ESTRUCTURAS DE CONTROL Y COMANDOS
3-> DEFINICION DE FUNCIONES Y PROCEDIMIENTOS
4-> VENTANAS Y GRAFICOS
5-> FICHEROS Y COMUNICACIONES
6-> MULTITAREA Y COMPILADORES
7-> TOOLKIT II

Este curso está muy comprimido y será necesario que os esforcéis un poco desde el primer capítulo, hemos quitado toda la paja que pudiera haber normalmente en cursos de este tipo, ya que con el primer capítulo podréis ser capaces de realizar pequeños programas.

ESQUEMA DE UN BUEN PROGRAMA

Un programa no puede ser un manojo de líneas escritas de cualquier manera lo más normal es que no funcione, pero si lo hace será difícil ampliarlo, entender cómo funciona o corregir algún error que seguro tendrá.

El esquema que a continuación os damos no es el único que se puede seguir, depende de gustos, pero es el más difundido y con el que conseguiréis una mayor estructuración de vuestros programas.

```
1000 REMark ==========================================
1010 REMark                  Titulo y version
1020 REMark                       Autor
```

```
1030 REMark                          Fecha
1040 REMark ==========================================
1050 !
1060 DEFINICION DE CONSTANTES
1070 DECLARACION E INICIALIZACION DE VARIABLES GLOBALES
1080 !
1090 PROGRAMA PRINCIPAL
1100 STOP
1110 !
1120 DEFINICION DE UN PROCEDIMIENTO O FUNCION
1130 FIN DE DEFINICION
1140 !
1150 DEFINICION DE OTRO PROC. O FUN.
1160 FIN DE DEFINICION
1170
1180 LISTAS DE DATOS
```

En este esquema aparecen cinco partes bien diferenciadas que son:

1-> Cabecera
2-> Definición de constantes y declaración de variables
3-> Programa
4-> Definición de una función o procedimiento
5-> Datos

Cada apartado tiene un número indeterminado de líneas y separamos uno de otro con una o varias líneas vacías (una línea vacía se introduce con su número y !).

PRIMER APARTADO: Cabecera.

No influye nada en el funcionamiento del programa, se puede omitir, pero sería tan garrafal de no poner cinco líneas, como vais a hacer un programa de 5000. Poned aquí toda la información necesaria para identificar el programa. Esto os será muy útil cuando posteriormente queráis localizar una determinada versión de algún viejo programa.

SEGUNDO APARTADO: Constantes y Variables

En SuperBasic no existen constantes, pero podemos suplirlas asignando un valor fijo a una variable, asignando un valor que ya no cambiaremos en el

transcurso del programa. Un ejemplo puede ser la constante g de gravedad a nivel del mar, que vale 9.8 m/s2.

Si nosotros tenemos que utilizar esta constante en un programa, podemos definirla declarando la variable g, y asignando el valor 9.8

En este apartado debemos asignar el valor correspondiente a todas las constantes del programa y declarar todas las variables del programa.

Aunque parezca un esfuerzo inútil, es muy beneficioso declarar todas las variables pues cuando nuestro programa crezca, podremos incorporar más sin miedo a que ya existiesen.

Además, si nosotros vamos a compilar nuestro programa seguramente el compilador (traductor a código máquina) nos exigirá algunos cambios, pues es más rígido que el interprete del SuperBasic, con el que habitualmente trabajamos.

TERCER APARTADO: Programa

Un programa no es mas que una sucesión de comandos y controles que manipulan datos sobre variables. Es aconsejable escribir un sólo comando por línea aunque se pueden poner varios si estos tienen mucha relación. El objetivo de esto es hacer que el programa no pierda legibilidad.

Aunque en SuperBasic existen los comandos GOTO y GOSUB es aconsejamos que no los utilicéis nunca. Se puede resolver cualquier problema de programación sin ellos, y el programa será más legible y versatil. Poned siempre la sentencia STOP al final del programa, aunque sea superfluo.

CUARTO APARTADO: Funciones y Procedimientos

De momento os diremos que definir una función o procedimiento es crear y añadir al SuperBasic, una función o comando nuevo. Como no importa el orden en que los definamos es preferible hacerlo en orden de lista así los localizaremos antes cuando haya muchos.

Una función es, para andar por casa, una palabra que nos devuelve siempre un resultado. Por ejemplo la raiz cuadrada que en SuperBasic se escribe SQRT (número) nos devuelve un resultado que es la raiz cuadrada de un numero. Un procedimiento no devuelve resultados sino que realiza alguna accion. Por ejemplo CLS es un procedimiento que borra la pantalla y no nos devuelve ningun resultado.

El aspecto entre una función y un procedimiento definidos por nosotros es casi el mismo, de hecho podemos definir funciones que hagan lo mismo que un

procedimiento definido. La diferencia es que una función nos devuelve un resultado que se espera que sea utilizado en una expresión o como parámetro para otra función o procedimiento (una expresión es por ejemplo lo que hay a la derecha del igual "=" de lo que sigue: A= 10*SQRT(12)+2, un parámetro es un dato que entregamos a una función o un procedimiento, el 12 anterior es un parámetro.

QUINTO APARTADO Datos

En un programa en SuperBasic podemos tener una lista de datos que podremos leer desde dentro del programa. No importa en que parte del programa se encuentren, No interfieren con el mismo, así que por claridad los colocaremos siempre al final

Para nosotros ahora un programa estará constituido como mínimo por los apartados 1,2 y 3, así que como mínimo tendrá una cabecera, una declaración de variables y el bloque del programa.
Vamos a continuación a escribir un programa que se ajuste a este formato
La temperatura habitualmente la medimos en grados Centígrados. Los físicos utilizan otra escala que tiene la misma unidad pero que está desplazada y se llama escala Kelvin. La fórmula de conversión es muy sencilla: K=C+273.15. El programa sería el siguiente:

```
100 REMark ********************************************
110 REMark         Conversión grados C a K
120 REMark         Por Lorenzo y Diego
130 REMark              30 3 88
140 REMark ********************************************
150 :
160 KC=273.15
170 C=0
180 K=0
190 :
200 CLS
210 INPUT " Temperatura en grados Centigrados: ";C
220 K=C+KC
230 PRINT " En grados Kelvin es: ";K
240 STOP
```

PRIMEROS COMANDOS DEL SUPERBASIC

Para escribir un programa necesitais algunos comandos para poder editar corregir y ejecutar dicho programa. Los más importantes son

COMANDO CLS

Este comando una vez ejecutado nos limpia la pantalla, pero dejando en memoria el programa y todas sus variables intactas, también podemos introducir este comando en el interior de un programa.

COMANDO EDIT

Con este comando editaremos cualquier linea que hubiéramos introducido anteriormente, en la que quisieramos modificar o corregir Su forma de manejo será EDIT n, donde "n" es el numero de linea que deseemos editar.

COMANDO NEW

Este comando nos borra todo el programa y sus variables, restablece diversas variables del sistema y nos borra la pantalla. También podemos usar este comando en el interior de un programa, consiguiendo el mismo efecto que el comando ABANDONAR del programa QUILL.

COMANDO LIST

Con él conseguimos un listado parcial o completo de nuestro programa. Si no le damos parametros nos da un listado completo que podemos parar con CRTL+F5. Si le damos parametros conseguimos listados parciales.

COMANDO RENUM

Con este comando conseguiréis renumerar parcial o totalmente el programa consiguiendo con ello dejar el espacio entre lineas que nosotros queramos, para introducir otras nuevas.

Basta teclear este comando para que se renumere todo el programa comenzando en la linea 100, y de 10 en 10 a partir de ella.

COMANDO RUN

Con este comando conseguiréis ejecutar vuestro programa, si le dais un parametro éste sera el número de linea por la que empezará, si no lo hará por la primera linea.

Con estos comandos podrás ya programar cómodamente, pero debes conocerlos bien, y sería bueno que los consultaras en el manual

VARIABLES SENCILLAS

Un programa tiene que procesar los datos que le damos y devolvernos resultados, consecuencia de ese proceso. Para que los datos puedan ser procesados tienen que ser de un tipo conocido por la máquina

Los dos únicos tipos de datos que manejamos en SuperBasic son números y caracteres, y a estos tipos los llamamos numérico y tipo alfanumérico. Estos datos el programa debe guardarlos en algún sitio, que lo denominaremos "variable", por lo tanto tendremos variables numéricas y alfanuméricas.

Guardar un dato en una variable, es asignarle un valor. La asignación se realiza con el símbolo '=', de la siguiente manera, si queremos guardar el dato 65987 en la variable numérica pts, debemos escribir pts= 65987, y si queremos guardar un nombre en la variable alfanumérica nombre$, debemos escribir nombre$="Jose Perez".

Estos datos permanecerán en estas variables mientras no asignemos unos nuevos o limpiemos el área de variables con el comando apropiado que ya explicaremos en otro momento.

En el Basic standard la asignación de la variable es precedida siempre de la palabra LET, que podríamos considerarla como un comando de asignación. Los ejemplos siguientes quedarían:

LET pts= 65987
LET nombre$="Jose Perez"

En otros lenguajes como PASCAL, FORTRAN o C, no se utiliza ninguna palabra para la asignación, y SuperBasic nos permite utilizarla o no indistintamente

COMANDOS DE ENTRADA Y SALIDA

Si nuestro programa tiene que aceptar una serie de datos para procesarlos y darnos una serie de resultados, tendrá que tener instrucciones que admitan la entrada de datos e instrucciones que admitan la salida de ellos. Estos comandos son PRINT para la salida e INPUT para la entrada

COMANDO PRINT

Con este comando podremos imprimir cualquier dato numérico o alfanumérico en

la pantalla. Este dato puede ser aportado directamente, mediante una variable o puede ser el resultado de evaluar una expresión.
Datos aportados directamente podrían ser:

```
PRINT 37
PRINT "EL PERRO LADRA"
```

Datos aportados por una variable serían:

```
a=27
p$="Patata"
PRINT a
PRINT p$
```

Datos aportados mediante una expresión serían:

```
a=27
PRINT 2*8
PRINT 2+a/3
```

Cada vez que ejecutemos una expresión de este tipo, el dato es impreso en la posición actual del cursor en la pantalla (El cursor viene a ser lo mismo que el bolígrafo, y tiene una posición en cada momento en la pantalla, así como el bolígrafo sobre el papel), y el cursor se coloca seguidamente al principio de la siguiente línea de la pantalla.

Si queremos hacer que el cursor permanezca a continuación del dato que hemos impreso, debemos colocar después de éste el símbolo ";".
Si queremos que permanezca en la misma línea, como antes, pero queremos que el siguiente dato quede tabulado en una columna, debemos colocar el símbolo "," en vez de ";".

Si queremos que avance hasta una determinada posición de la línea, debemos utilizar la función TAB. Supongamos que el cursor avance hasta la posición 40 de la línea; entonces debemos poner 'TAB (40);' en vez de ";".
Si tenemos que sacar diversos datos por pantalla y las sentencias PRINT dato, están seguidas podemos agruparlas en un solo PRINT. Los datos aparecen uno detras de otro después del comando PRINT y separados por un símbolo que pueden ser los anteriores teniendo el efecto ya explicado o el símbolo "\" (barra invertida que se obtiene con CRTL+8), que coloca el cursor al principio de la línea siguiente.

Ejemplos:

```
PRINT 23456;" PTS"
PRINT "RESULTADO : "&23456;" KILOMETROS"
```

COMANDO INPUT

Con este comando introduciremos datos numéricos o alfanuméricos que serán asignados a variables numéricas o alfanuméricas. Cuando un programa se está ejecutando y llega a una instrucción INPUT, el programa esperará a la introducción de un dato, y lo asignará a una variable para su posterior uso

Ejemplo:

```
INPUT años
INPUT nombre$
```

La introducción del dato se hará desde el teclado y la veremos en pantalla en la posición actual del cursor de texto. Si queremos que un programa nos pregunte nuestro nombre escribiremos lo siguiente:

```
PRINT "Dame tu nombre ";
INPUT nombre$
```

El procedimiento INPUT puede imprimir el mismo los mensajes que preceden a la entrada del dato. Así el ejemplo anterior quedaría:

```
INPUT "Dame tu nombre  ";nombre$
```

El procedimiento INPUT admite la misma construcción que PRINT, solo que las variables serán preguntadas. Si una variable forma parte de una expresión, no es preguntada, si no que la expresión es evaluada e impresa.

Así que si una variable queremos que no sea preguntada si no impresa, deberemos hacer que aparezca en una expresión, por ejemplo, poniendo la variable entre paréntesis. Ejemplo:

```
INPUT (A);B   . .   ,imprime el dato de A y nos pregunta el de B
```

[illegible]

En un programa nos podemos encontrar con que cierta acción la tenemos que realizar sólo si se cumple cierta condición o que la tenemos que repetir cierto número de veces o que la tenemos que repetir hasta que se cumpla cierta condición. Cada uno de los casos anteriores constituye una estructura de

programación. Hay más, tu mismo puedes deducirlas, son las estructuras con que pensamos.

Cada lenguaje de programación incluye una colección de ellas. La capacidad de estructuración de un lenguaje dependerá de las estructuras de control que permita.

La estructura que vamos a ver, nos permite repetir infinitas veces una acción, lo que denominamos bucle, que en este caso es infinito. Si bajo ciertas circunstancias se termina dicha repetición podemos llamarlo bucle con salida condicionada.

En SuperBasic para hacer que una secuencia de instrucciones sea un bucle, hay que ponerlas entre dos palabras que marcan el principio y el fin de la secuencia, dando además nombre al bucle. Estas dos palabras son REPEAT y END REPEAT, poniendo el nombre a continuación del REPEAT. Como ejemplo vemos un bucle que se llama "tonto" y escribe muchos "hola" en la pantalla.

```
100 REPEAT TONTO
120      PRINT "HOLA"
130 END REPEAT TONTO
```

REPEAT TONTO indica el principio de un bucle que se llama TONTO. END REPEAT TONTO, indica el fin de la secuencia que constituye dicho bucle. Este bucle se repetirá indefinidamente y si quieres parar el programa tendrás que pulsar CTRL + SPACE (Barra espaciadora). Si queremos que la ejecución del bucle se termine alguna vez, tendremos que insertar en él, un control que bajo ciertas condiciones produzca la salida del bucle, este control tiene el siguiente formato:

IF condición THEN EXIT nombre del bucle

Donde condición es normalmente una comparación, pero también puede ser una variable, un expresión o una función.
Comparaciones pueden ser:

5=5 (que se cumple)
5<7 (que también se cumple)
A<7 (que se cumple si el número contenido en A es menor que 7

Los símbolos "=" y "<" son comparadores. Disponemos de los siguientes comparadores:

= es igual a
< es menor que

```
>          es mayor que
<=         es menor o igual que
>=         es mayor o igual que
<>         es distinto de
```

Vamos a hacer que el bucle TONTO escriba sólo diez "hola"

```
100 N=1
110 REPEAT TONTO
120       PRINT "HOLA"
130       IF N=10 THEN EXIT TONTO
140       N=N+1
150 END REPEAT TONTO
```

Apartir de ahora escribe siempre todas las instrucciones del interior de un bucle varios espacios más a la derecha (identar). Con ello los programas son mucho más facil de seguir.

PROGRAMAS Y EJERCICIOS

En todos los capitulos vamos a proponer los programas y a resolveros algunos En este capitulo son muy faciles, pero si no empiezas a hacerlos ahora más vale que no te pases el curso y sigas siendo un consumidor de programas hechos por otras personas

El siguiente programa cuenta desde uno hasta el número que tú le digas y los imprime en la pantalla Como ves se ajusta al formato que te pedimos que utilices. En la zona de declaración de variables aparecen dos variables declaradas que son N y M. M contendrá el tope hasta el que queremos contar Y N la utilizaremos como variable contadora. El programa empieza por preguntarte el tope y almacenarlo en M. Luego inicializa N a 0; seguidamente entramos en el bucle contador del programa en el que incrementamos la variable N, la imprimimos y comprobamos si sobre pasa a M Como ves el programa siempre imprimira el 1 aunque tú le des tope menor que 1 Cambiando adecuadamente el orden de estas instrucciones podras conseguir que no se imprima el 1 si le damos un tope menor

```
100 REMark ==========================================
110 REMark          PROGRAMA CONTADOR
120 REMark    Escrito por: Diego y Lorenzo
130 REMark          fecha: 11/04/88
140 REMark ==========================================
```

```
150 '
160 N=0
170 M=0
180 '
190 CLS
200 INPUT "HASTA DONDE CUENTO? ";M
210 N=0
230     N=N+1
240     PRINT "==) ";N
250     IF N>=M THEN EXIT CONTAR
260 END REPeat CONTAR
270 STOP
```

Este programa te pregunta tu nombre y con mucha educación te saluda. Para ello solo necesita una variable: nombre$. Lo primero que hace es borrar la pantalla. A continuación te pregunta el nombre, para ello el procedimiento INPUT imprime en la pantalla un mensaje que te informa acerca de la pregunta ("¿Como se llama? "). Observa que el mensaje tiene un espacio al final. Este espacio sirve para separar el texto del mensaje de los datos que introduciremos. Esta misma precaución hay que tenerla en el PRINT después de Sr. hay un espacio.

```
100 REMark ************************************************
110 REMark            PROGRAMA SALUDO
120 REMark     Escrito para Diego y Lorenzo
130 REMark            fecha: 11/04/88
140 REMark ************************************************
150 '
160 nombre$=""
170 '
180 CLS
190 INPUT "¿Como se llama?  ";nombre$
200 PRINT " Hola, Buenos dias Sr. "nombre$
210 PRINT "¿Que tal esta usted?"
220 STOP
```

El siguiente programa nos imprime una tabla de cuadrados y cubos de números entre dos números dados. Declaramos sólo dos variables y la variable en la que almacenamos el extremo inferior la utilizaremos como variable contadora.

Después de preguntarnos los límites el programa imprime unos rótulos para la lista que deberemos procurar que queden alineados con las columnas de la tabla. Este programa no es más que el programa contador modificado para el fin que hemos querido.

```
100 REMark ==========================================
110 REMark          PROGRAMA TABLA DE CUADRADOS
120 REMark    Escrito por: Diego y Lorenzo
130 REMark          fecha: 11/04/88
140 REMark ==========================================
150 :
160 num=0
170 fin=0
180 :
190 CLS
200 INPUT "Primer número de la tabla: ";num
210 INPUT "Ultimo número de la tabla: ";fin
220 CLS
230 PRINT
240 PRINT ,"Numero          Cuadrado        Cubo"
250 PRINT ,"------------------------------------"
260 REPeat tabla
270     PRINT ,num,,num*num,,num*num*num
280     num=num+1
290     IF num>fin THEN EXIT tabla
300 END REPeat tabla
310 STOP
```

Este programa simula numéricamente la caida libre de un objeto. En el introducimos la definición de una constante g=9.8, y luego declaramos tres variables físicas: la altura, la velocidad y el tiempo.

El programa empieza por preguntarnos la altura desde la que lanzamos el objeto y la velocidad inicial. Si la velocidad inicial es 0 significa que simplemente lo soltamos, si es positiva lo lanzamos hacia arriba y negativa que lo lanzamos hacia abajo.

A continuación el programa imprime en pantalla una tabla muy similar a la del programa anterior en la que tenemos la altura y velocidad del objeto en diferentes instantes de su caida. Los incrementos de tiempo están ajustados a un segundo.

```
100 REMark ==========================================
110 REMark              Simulación de caida libre
120 REMark              Por Diego y Lorenzo
130 REMark                    fecha 14/04/88
140 REMark ==========================================
```

```
150 :
160 g=-9.8
170 h=0
180 v=0
190 t=0
195 :
200 CLS
210 INPUT"Altura de lanzamiento: "sh
220 INPUT"Velocidad inicial(+ hacia arriba): "sv
225 t=0
230 PRINT
240 PRINT"       Tiempo          Altura          Velocidad"
250 PRINT"  ============================================="
260 REPeat caida
270       IF h<0 THEN EXIT caida
280       PRINT,t,,h,,v
300       h=h+v+g/2
301       v=v+g
310       t=t+1
320 END REPeat caida
330 PRINT"  ============================================="
340 PRINT"            Plofff"
350 STOP
```

El programa carta personalizada es un ejemplo del partido que se le puede sacar al QL si se tiene un poco de interes. Este programa nos pregunta un nombre (nos podria preguntar más cosas), y luego nos imprime en pantalla una carta dirigida a esa persona.

Si en vez de imprimirla en la pantalla la hiciéramos por la impresora muy bién nos podria servir para imprimir circulares o cartas a clientes. Si tienes mucho interes en hacerlo ya te podemos adelantar como hacerlo aunque el porque te los explicaremos más adelante.

Introduce la siguiente linea:

```
205 OPEN #3,SER1
```

Y luego sustituye todos los PRINT del programa por:

```
PRINT £3;
```

Con ello conseguirás que tu carta salga por la impresora. Con estos escasos conocimientos podrás conseguir un alto rendimiento de tu equipo.

```
100 REMark ==========================================
110 REMark            Carta Personalizada
120 REMark            Por Lorenzo y Diego
130 REMark              fecha 14/04/88
140 REMark ==========================================
150 '
160 nombre$="
170 '
180 CLS
190 INPUT" Nombre del destinatario: ";nombre$
200 CLS
210 PRINT "                                                  Cerrica Aragonesa S.A."
220 PRINT "                                                  Polígono Los Angeles"
230 PRINT "                                                    Zaragoza 50022"
240 PRINT
250 PRINT "     Distinguido Sr. "nombre$";"
260 PRINT
270  PRINT"      Somos una joven empresa en el mercado y ofrecemos nuestros pro
ductos con una elevada calidad así como una cuidada elaboración, manteniendo sin
embargo unos precios muy competitivos y que suponemos serán de su interes."
280 PRINT"Adjuntamos en esta carta una lista de nuestros productos así como su
s respectivos precios a nivel de distribuidor."
290 PRINT"Sin otro particular y esperando que nuestros productos sean de su ag
rado, nos despedimos de Usted, a la espera de sus noticias."
300 PRINT
310 PRINT
320 PRINT"                                                  Atentamente,"
330 PRINT
340 PRINT"                                               Eduardo Arriestegui Ternasco"
350 PRINT"                                                  Director Comercial"
```

Diego Alcalá y Lorenzo Alluda

LIBRERIA

Estimados socios :

Para los que no os habeis enterado todabia, deciros que se han formado ya hace tiempo, varios grupos locales, no teneis más que ver las revistas para saber que ya estan funcionando a toda marcha, asi que los que en vuestras respectivas localidades disponeis de grupos locales, no dudeis en dirigiros a sus respectivos encargados, para pediros todo tipo de información, colaboración, ayuda y sobre todo saber que disponen toda la libreria de programas en disco, para los que querais obtener algún tipo de programa que os interese de manera particular.

Los que no conteis con algun grupo local, en primer lugar deciros que os animeis para formarlo, no teneis nada más que comunicarnoslo y os daremos toda información y ayuda para realizarlo, y en segundo lugar saber que estamos para ayudaros en lo que sea necesario y ofreceros toda nuestra colaboración. No dudeis en pedirnos cualquier cosa, estamos para vuestro servicio, animo y adelante.

Para los que tubia tengais pendientes de recibir tanto microdrives como discos o cualquier otro coso mandar una carta al apartado, indicando seccion 'Pendientes', apartando fotocopia de giro realizado.

Nuevo programa para libreria,

85.- JUEGO FORTH Sectores: 54 FORTH Didactico

Fi Juego_Num_fth, Array_fth, Juego_conversión, J_Num

Este juego nos adentra en las utilidades del Forth, que tan magnificamente nos aporta nuestro formidable socio Salvador Merino, un genio del Forth como ya podeis comprobar. Para mas información leer su formidable cuarso.

86.- TRON Sectores: 22 SuperBasic/Compilado Juego

Fi Boot_juego, qlib_run, juego2_obj, juegojoy

Este juego de laberintos, nos introducira a los escenarios de la pelicula Tron, muy parecido a versiones hechas para el Spectrum. Tenemos varias formas de funcionamiento, normal, joysticks, compilado, elegir.

SUMARIO

1.- PORTADA "MARILYN".
2.- INFORMACION DEL CLUB.
3.- EDITORIAL.
5.- DESPEDIDA.
6.- CORREO DE LOS SOCIOS.
8.- NOVEDADES.
9.- STRIP POKER.
10.- PREGUNTAS DE LOS SOCIOS.
13.- TRUCOS.
14.- COLABORACIONES DE LOS SOCIOS.

14 * CURSO DE FORTH (IV, V, VI).
22 * ORDENACION Y BUSQUEDA DE DATOS (I)
26 * CONJUNTOS, SETS . . . Y SuperBasic
36 * ENCUESTA Y NAVEGACION

38.- COMENTARIO DE PROGRAMAS

38 * ENSAMBLADORES Y LIBROS.
41 * SPEEDSREEN.
43 * SUCCESS CP/M.
45 * DIGITAL C.
48 * ALIEN HIJACK
49 * MEGA-TOOLBOX
52 * QI EXPERT

55.- COMENTARIO DE LIBROS

55 * LENGUAJE C
55 * LENGUAJE LISP

57.- CURSO DE SuperBasic
71.- LIBRERIA
72.- SUMARIO

D.L.Z. 162 - 87